NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

PLANNING CONSIDERATIONS FOR SECURE NETWORK PROTOCOLS

by

Philip R. Barlow

March 1999

Thesis Advisor: G. M. Lundy

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	March 1999	Master's Thesis

4. TITLE AND SUBTITLE
PLANNING CONSIDERATIONS FOR SECURE NETWORK PROTOCOLS

5. FUNDING NUMBERS

6. AUTHOR(S)
Barlow, Philip R.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)*

An attempt is made to provide the reader with an appreciation for incorporating basic security services within a network protocol (e.g., multicast). Security incorporated within a network design is an increasingly common requirement that users are levying upon network implementations (military and commercial). Network security implementations evoke a myriad of abstractions, technologies and other related issues that can overpower a reader and cloud the topic with details. This thesis is intended to assist readers achieve an overview and background of the varied subject matter network security implementation necessitates. Essential services are introduced and discussed to provide an understanding of what constitutes an adequate and efficient security implementation. Related infrastructure (key distribution/ management) requirements needed to support network security services are examined. The thesis concludes by identifying tactical user network requirements and suggests security issues to be considered in concert with network implementation.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Planning Considerations For Secure Network Protocols	151
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

Approved for public release; distribution is unlimited.

PLANNING CONSIDERATIONS FOR SECURE NETWORK PROTOCOLS

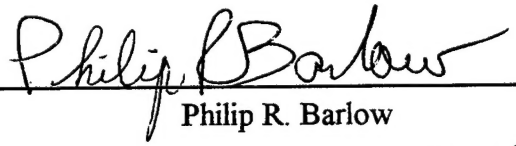
Philip R. Barlow
M.S., San Diego State University, 1976

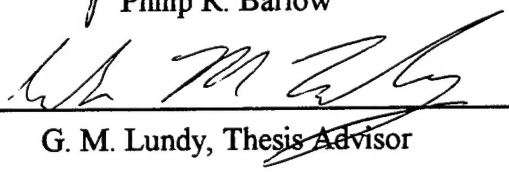
Submitted in partial fulfillment of the requirements for the degree of

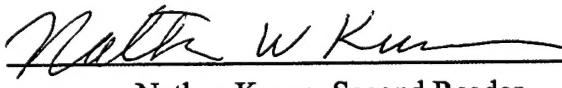
MASTER OF SCIENCE IN SOFTWARE ENGINEERING

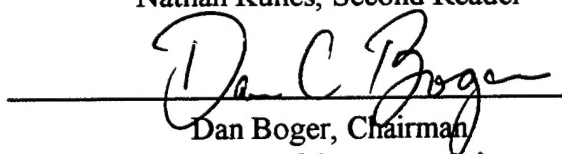
from the

NAVAL POSTGRADUATE SCHOOL
March 1999

Author: 
Philip R. Barlow

Approved by: 
G. M. Lundy, Thesis Advisor


Nathan Kunes, Second Reader


Dan Boger, Chairman
Department of Computer Science

ABSTRACT

An attempt is made to provide the reader with an appreciation for incorporating basic security services within a network protocol (e.g., multicast). Security incorporated within a network design is an increasingly common requirement that users are levying upon network implementations (military and commercial). Network security implementations evoke a myriad of abstractions, technologies and other related issues that can overpower a reader and cloud the topic with details. This thesis is intended to assist readers achieve an overview and background of the varied subject matter network security implementation necessitates. Essential services are introduced and discussed to provide an understanding of what constitutes an adequate and efficient security implementation. Related infrastructure (key distribution/ management) requirements needed to support network security services are examined. The thesis concludes by identifying tactical user network requirements and suggests security issues to be considered in concert with network implementation.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS/SYMBOLS/ABBREVIATIONS

ACK	Acknowledgment
AFATDS	Advanced Field Artillery Data System
AH	Authentication Header
AS	Authentication Server
ATM	Asynchronous Transfer Mode
	Automatic Teller Machine
BER	Bit Error Rate
C2	Command and Control
CA	Certification Authority
CBC	Cipher Block Chaining
CBT	Center Based Tree
CDMA	Code Division Multiple Access
COTS	Commercial Off-The-Shelf
CPU	Central Processing Unit
DAC	Data Authentication Code
DISN	Defense Information Systems Network
DES	Data Encryption Standard
DMS	Defense Message System
DR	Designated Receiver
ECC	Elliptic Curve Cryptosystem
ESP	Encapsulated Security Payload
FIPS	Federal Information Processing Standard
GCCS	Global Command and Control System
GMR	Goldwasser/Micali/Rivest
HQ	Head Quarters
IAS	Intelligence Analysis System
IDEA	International Data Encryption Algorithm
IEC	International Electromechanical Commission
IGMP	Internet Group Management Protocol
IP	Internet Protocol

IPng	Internet Protocol next generation (version 6)
IPv6	Internet Protocol version 6
ISAKMP	Internet Security And Key Management Protocol
ISO	International Standards Organization
IT 21	Information Technology for the 21st century
ITU	International Telecommunications Union
IV	Initialization Vector
JMCIS	Joint Maritime Command Information System
JTF	Joint Task Force
KAS	Kerberos Authentication Server
KDC	Key Distribution Center
KTC	Key Translation Center
LAN	Local Area Networks
MAC	Message Authentication Code
MAGTF	Marine Air Ground Task Force
MAN	Metropolitan Area Network
MCSSC2	Marine Combat Support Command and Control
MD	Message Digest
MDC	Modification Detection Code
MEB	Marine Expeditionary Brigade
MEF	Marine Expeditionary Force
MEU	Marine Expeditionary Unit
MIC	Message Integrity Code
MLS	Multi-Level Security
NACK	No Acknowledgment
OSI	Open Standards/Systems Interconnection
OSPF	Open Shortest Path First
OWHF	One Way Hash Function
PGP	Pretty Good Privacy
RFC	Request For Comments
RIPMED	European RACE Integrity Primitives Evaluation
RMTP	Reliable Multicast Transport Protocol
RSA	Rivest/Shamir/Aldeman

SAFER	Secure And Fast Encryption Routine
SBT	Sender Based Tree
SHA	Secure Hash Algorithm
S/N	Signal to Noise
TCO	Tactical Combat Operations
TDN	Tactical Data Network
TTP	Trusted Third Party
VTC	Video Teleconferencing
URL	Uniform Resource Locator

I. INTRODUCTION

This thesis is intended to provide an organized and understandable body of information for the network engineer and others who are required to support information security in open/sensitive environments. It is hoped that this body of information will assist the reader to reduce the time and effort of performing literature searches, or endless "surfing" through Web sites. This research is designed to:

1. Discuss Security Concepts, Issues, and Implementations Important to multicast protocols (e.g., RMTP)

Provide a basic conceptual understanding of reliable multicast transport protocols (RMTP) and associated security mechanisms. This discussion will serve the reader as a preparation for more technical discussions. The issues discussed will aid the reader in forming better problem solutions. Security strategies available to RMTP implementations will be discussed so that security cost/benefit implications can be more fully appreciated.

2. Define Security Terminology From the Open Literature

Important terms gathered from review of open sources will be defined and listed in the Glossary. This Glossary is intended to assist the reader communicate with other technical professionals within the discipline.

3. Provide References to Additional Sources of Information on Network/Security Topics

The reader may face security requirements that merit further consideration and research, in order to achieve a successful implementation. The List of References contain resources that provide the background for the topical treatments within this thesis. They contain further information that can provide the reader with more in-depth details. Network security-related Web are contained in Appendix A. These web pages provide the reader information basic to the field of network security, as well as, latest commercial implementations.

The discussion will identify essential security services and provide an explanation of what services are provided to network users.

A. SECURITY SERVICES

In today's world, network security has taken on the trappings of a sleeping giant waking up, no longer content to be ignored, but confronted head on. And the more information bandwidth that technology makes available to network users, the more user demand for bandwidth grows. Security services and bandwidth appear to compete with each other for available information payload space (packets/datagrams) on open data networks. The rapid

advance of hardware technology has created expectations that there will always be a solution for today's need of greater security services while maintaining or, even increasing information throughput.

Hardware's ubiquitous capacity to satisfy user requirements for security and bandwidth does have its limit and associated price. The intent of this paper is to uncover the limitations and implications associated with various concepts and methods that provide network users with the degrees of security they require, vice desire. Security issues and alternatives will be investigated with suggested implementations in multicast environments.

The concept of secure (i.e., confidential) communications is not new and instances trace back to the Egyptians, who displaced the order of hieroglyphic letters. This reordering was an implementation of a simple substitution, or shift cipher. Julius Caesar is reported to have used a three-letter shift (to the right), called a Caesar cipher, to communicate secretly with his army. Encryption began with these simple substitution algorithms and has evolved into complex computer algorithms that consume increasing amounts of central processing unit (CPU) resources. Encryption of various strengths can provide the user with an appropriate level of data confidentiality. The

level of confidentiality assures communicating parties that the information they pass to each other will not become known, or available to individuals, processes, or entities that do not have an authorized need to know. Another security service even more basic than confidentiality is authentication.

A common example of an authentication protocol (a rule based communication exchange) is the sentry guarding his post (shown in Figure 1). An unidentified soldier

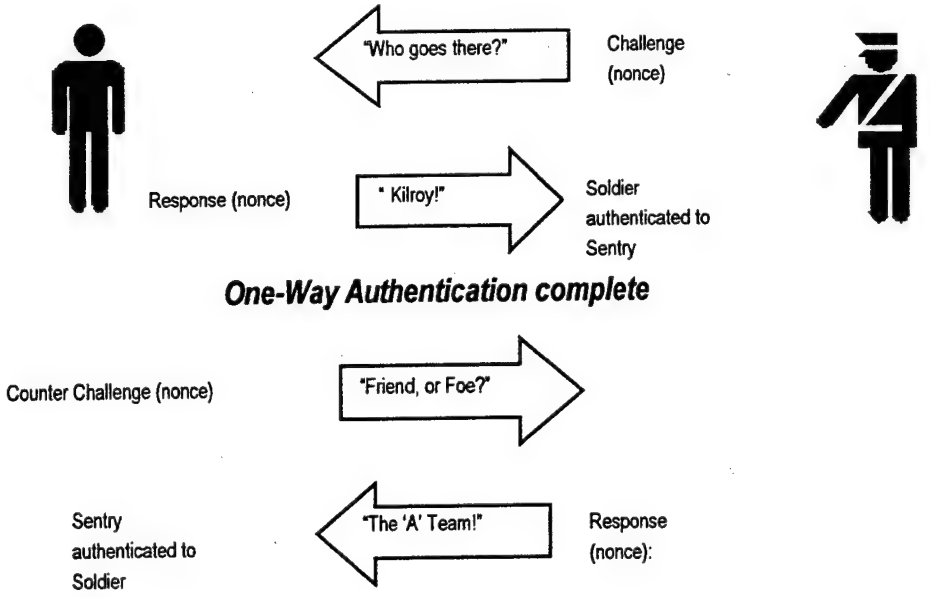


Figure 1. Example Authentication Protocol

approaches, the sentry sends a challenge, "who goes there". If the soldier recognizes the challenge and replies

correctly ("Kilroy") to the sentry, the protocol is satisfied, and the soldier's identity is authenticated to the sentry as a colleague (or, could he be an intruder who has covertly obtained the soldier's authentication code?). This is an example of one-way authentication. The sentry assumes the authenticated soldier is friendly, because the soldier has knowledge of a secret (an authentication code) shared with the sentry. The soldier, however, has no information about the sentry's identity (whose army is the sentry in?). A correct response to a counter-challenge, from soldier to sentry, is required to authenticate the sentry to the soldier.

Such a protocol provides a two-way (mutual) authentication service to the parties involved in this communication exchange at a two-fold cost in the number of required protocol exchanges. User requirements will indicate if the cost (additional communication exchanges) is justified. Most military implementations require mutual authentication services. Authentication is the basic security service, i.e., knowledge of whom one is in communication with (I am exchanging data with my colleague, Joe).

The second basic security service for discussion is data integrity. Authentication services go naturally with

data integrity services. It makes little sense to authenticate a party and be subject to undetected reception of deliberately tampered data. Data integrity assures a party that information received is tamper-free (i.e., information has not been subject to unauthorized alteration, destruction, or corruption). The threat of data tampering can not be eliminated, but tampered data can be detected using encoding and unkeyed hashing schemes. These techniques employ algorithms (mathematical, or other) that assure correct order and accuracy of the received information.

Message digests (MD) are used to ensure data integrity, and are constructed from one-way hash function algorithms. Hash functions readily calculate a unique fixed length bit string (the hash value) from the associated input message block with minimal computational burden. A particular hash value is unique to the message block used to generate that hash value. When hash functions are correctly constructed, it is effectively impossible to reproduce the message block from the corresponding hash value. Message digests (and hash values) are typically smaller than the message blocks they are generated from. A secret key value combined/appended with the message block and then passed through a one-way hash function produces a message

authentication code (MAC, or data authentication code [DAC]). MACS/DACS are techniques used to authenticate and verify integrity of file transfers (Schneier, 1996). This duality of MACs alludes to the natural association of authentication and integrity services.

This paper focuses on implementing three security services (authentication, integrity and confidentiality) into network environments (e.g., multicast, a single source transmitting information to a designated group of network users). Other security services are available, but discussion will focus upon the above mentioned services.

Other security services, such as, non-repudiation, access control, traffic analysis, denial of service, etc. provide useful functions that are well represented in the literature. They add refinements and build upon the basic network protection users increasingly expect. Non-repudiation (of origin) ensures that the sender can not deny authoring a particular document, much like the seal of a Notary public. And similarly, non-repudiation (of delivery) verifies that the receiver can not deny receipt of documents (Ford, 1994).

Access control ensures that only authorized personnel can make use of specified network resources. Access control can either restrict physical availability to network

resources (keyboard, computer, etc.), or employ access control lists (ACL) containing names, passwords, or other forms of user identification with which to control and manage network resources.

Traffic analysis occurs when an entity passively monitors activity on network segments to gain unauthorized information based upon heuristics of the traffic. Denial of service is active interference with network operation by obstructing network availability to legitimate users. These topics are not essential to basic network function and their exclusion from subsequent discussions will not detract from the basic concepts of secure network (multicast) environments.

B. PROTOCOL STACK AND SECURITY

This section overviews security services relative to a layered communications network protocol model. Most discussions of layered protocol architectures use the Open Systems Interconnection (OSI) Basic Reference Model (ISO/IEC 7498-1). It is assumed that the reader has a familiarity with the concepts of the seven OSI layers (Tanenbaum, 1996). The ensuing discussion assesses benefits of security services with respect to the layer in which the service is located.

Security services can be implemented in one specific, or several network protocol layers. Services rendered at each protocol layer add to the overhead that is carried along with the payload, datagram, message data unit, packet, frame, etc. Strategic security service placement at the most advantageous protocol layer reduces redundancy, and increases the transmission efficiencies of the network protocol suite.

Implementation of security services in the upper layers (session, presentation and application) rest predominately upon the application layer. No security services are provided in the session layer, and only those facilities that support security services implemented in the application layer are contained within the presentation layer. Manifestation of all upper layer security services appears in the application layer, and all upper layer security services are provided at this layer. The application layer is the only layer where integrity/confidentiality of selective data fields and non-repudiation services are provided (Ford, 1994).

Lower level network protocol security implementations have the advantage of lower equipment/operation costs, and this fuels the debate (in the literature) about which layer to implement security features. The only security services

available at the physical layer are connection confidentially (based upon the data unit) and/or traffic flow confidentially. These security services relate to mechanisms applied to the physical media for traffic flow, and/or via encryption for connection confidentiality of the bit stream. Cable and fiber transmission media can be shielded and physically protected within a conduit. Free-air transmissions can use spread spectrum (code division multiple access[CDMA}/direct sequence, or frequency hopping) techniques. Security requirements of each network segment will influence the method that is employed (e.g., physical media protection, or message content protection, via encryption). Encryption directly adds to the processing overhead cost of moving bits. Transmission media security techniques add to the hardware requirement of each protected network segment, but these equipment costs are generally low. Associated administrative network operating costs can greatly increase the independent management of each separately protected network segment. Security techniques, similar to those discussed for the physical layer, can be employed at the data link layer to provide connection, or connectionless (data unit) confidentially. The difference is that security services are provided at the frame level

(to maintain frame confidentiality), vice the bit level (Ford, 1994).

· More security services can be provided, to varying degrees, by the network layer. Network layer security services supported are authentication, integrity (except for selective data fields, and their recovery), confidentiality (except for selective fields) and access control. The network layer is the first network protocol layer that has the capability to contain most of the basic security services demanded by today's network users. These security services can generally be provided more economically at this layer than at the transport layer. However, the option for multi-level security (MLS) implementation is obtained in the transport protocol layer (Ford, 1994).

The final "lower layer" network protocol is the transport layer. This layer is thought by some to be the "heart" of the protocol hierarchy (Tanenbaum, 1996). Possibly because this layer can transparently provide the user with reliable end-to-end connectivity (source to destination[s]) across all networks. And as can be expected, authentication, integrity, and confidentially (the core security services) can all be provided (except for selective fields) within this layer. The transport layer can satisfy secure communication requirements through

untrusted intervening networks between sender and receiver end-systems, independent of the user application, and for all traffic on a connection. The debate between network vice transport layer security service implementation remains unresolved, and standards have been generated that support each view (ISO/IEC 11577, and ISO/IEC 10736, respectively).

For more detailed information about these layer related security service topics, the reader can refer to descriptions given in the OSI Security Architecture ISO 7498-2 and ITU-T Recommendation X.800. Discussions in Chapter V center on transport layer security implementations.

C. MULTICAST PROTOCOLS

Traditional data transmission protocols unicast data packets from one sender to a single receiver. When a sender wishes to transmit data to multiple receivers, the natural extension is to create additional unicasts, equal to the number (n) of intended receivers potentially repeating the same data (n-1) times. This approach (unicast) would not present any problems if the communication channels (media) could provide ample capacity. Adequate transmission capacity is not a prudent or realistic assumption. The Gulf War of the early 1990's dramatically pointed out that there was not enough bandwidth to satisfy every user's needs.

Multicast protocols reduce demands on bandwidth by consolidating data transmissions on network segments common to multiple receivers and pass only a single message, vice n copies of the same message. This type of transmission protocol is essential for scarce and over tasked tactical data networks (TDN) where multiple recipients receive the same data.

When referring to network user data requirements, the underlying assumption throughout this paper will be of the of TDN implementation types reported by Petitt, 1996. These tactical internets have requirements that differ from their commercial counterparts. For example, TDNs can simultaneously have mixed media and segments with greatly differing capacities (bandwidth & data rate), poor reception quality (bit error rates[BER]/noise/disruption), and dynamic topology configurations (network membership) with associated secure data requirements. These design constraints of these networks not only drive the choice of network transmission protocols, but also the types of security solutions.

Two reliable multicast protocols, from the many multicast protocols described in the literature (i.e., reliable multicast transport protocol [RMTP] & Internet protocol version 6 [IPv6]), and general considerations unique to wireless protocols, will provide a reference for

discussion on network security implementation. Each protocol has features applicable to TDNs that compounds the associated security service requirements implicit in deployed TDNs.

RMTP is AT&T's proprietary transport layer (4) sender based tree (SBT) delivery system that adds guaranteed undamaged packet delivery (reliable), accommodates scalable dynamic organization of receivers (groups), and allows for diversity of receiver segment bandwidth and processing power (Lundy et al, 1996). These services are generated within this protocol vice added over basic network layer (3) multicast routing protocols. RMTP employs a hierarchical SBT delivery scheme with subtrees emanating from uniquely specified receivers (designated receivers[DR]). Messages are cached at the sender and the DRs to provide an error recovery mechanism for lost/damaged packets, late group membership join, or allow a slow receiver to obtain missing data. Windowed flow control avoids congestion caused by overrunning slow receivers, or low bandwidth segments. RMTP is a bandwidth efficient scheme to reliably deliver data to users/applications that can accommodate delivery delays (latencies).

The IBM Japan/NTT RMTP (version 1) is a center-based tree (CBT) (center specific, or core driven) delivery system

primarily intended for cabled connectivity. Reliable delivery of large data files is based upon retransmission (unicast, or multicast selective repeat), handshake session control, and sender notification (initiated by individual receivers) of transmission success. The reliability service (of this RMTP version) is added over the basic network layer (3) IP multicast routing protocol (i.e., the Internet group management protocol [IGMP]). The commercial implications of this data delivery system include delivery of: electronic newspapers, shopping catalogs, music, videos (on demand), software, and enterprise software updates. Flow control is an option that is based upon an experimental dynamic control mechanism allowing the sender to adjust the data rate according to the congestion status (indicated by a block loss ratio). This multicast method transmits data all at once to the multicast receivers. After a time out, receivers that successfully receive all the data packets send ACKS, and those receivers that require retransmission of data packets send NACKS. The NACKS identify those packets that the receiver is requesting. When few receivers request a particular data packet, RMTP can switch to a unicast mode to reduce unnecessary retransmissions. Modeling and tests on LANS with large-scale receiver simulators have established the workability of the

retransmission scheme employed by RMTP (T. Shiroshita, et al, 1996).

IPv6 or, next-generation IP (IPng) is the new emerging Internet protocol standard that is gaining pervasive acceptance in the international Internet. Vendors such as Digital Equipment Corporation (i.e., Compaq), Apple, Hewlett-Packard Novell, and Sun Micro systems have started to deliver workstations and servers with IPv6 installed. Ipv6 is a ground up design that provides increased network throughput performance, scalability, security, ease-of-configuration and network management functions, as well as an increased address space (128 bits). It uses source based Open Shortest Path First (OSPF) network layer routing protocols to achieve high network performance. Network layer security services (authentication, integrity and confidentiality) have been built into IPv6 to provide secure interoperable messaging for all Internet users (Bay Networks, 1997).

Mobile (wireless) communications present a more dynamic set of user requirements to be satisfied. Host limitations on power consumption, data storage processing speed, lower capacity and noisy transmission channels, and fluid multicast membership combine to generate new levels of complication for a multicasting protocol. IBM Japan/NTT

have developed a successor version of their multicast protocol, called RMTP (version 2). This protocol has been designed especially for mobile (wireless) users. Terminal authentication, data encryption, quick restart of interrupted transmissions, and transmission speed control functions have been added to the previous protocol (RMTP [version 1]) providing high data reliability to mobile users (Shino, et al, 1997).

D. SYNOPSIS

1. Focus

This thesis emphasizes the security services that are essential to users in tactical land and naval multicast environments. The majority of the discussions will focus upon the relationship of security services in a cable-connected Internet multicast environment. Limited discussion will describe aspects of mobile (i.e., wireless) multicast protocol security services.

2. Limitations

There are volumes of information available on the topics discussed in this thesis. The focus has been on the many and diverse security related topics, such as, network transmission protocols, encryption algorithms, authentication and key management protocols, data integrity features, and user data requirements. The reader should

consider this work a sample of the available information and make good use of the references in the back of this document.

3. Assumptions

The intent of this thesis is to give the reader a background in the concepts of the protocol layers and related security features, and to inform the reader about security issues and suggest implementation considerations in tactical and Naval data networks. Terminology and concepts are introduced in a manner that will allow the reader to become familiar with the basic issues that are of primary concern to network security implementation.

II. MAJOR ISSUES

This chapter provides an overview of the major issues (essential security services/tools available, key management techniques, and suggested multicast security implementations) addressed in this thesis. The following chapters (chapters III, IV and V) expand upon these issues, with a final chapter (VI) to discuss conclusions and recommendations.

A. ESSENTIAL (NETWORK) SECURITY SERVICES (CHAPTER III)

There has been much discussion, and there will be even more, as strong encryption technology becomes available to more network users. The fundamental security question is which security services are essential, and which ones are nice-to-have. The answer is not as straightforward as the mathematics that drives the encryption algorithms. Consideration for individual user-weighted blending of: data value, security service cost, user implementation acceptance, and other performance tradeoffs will influence the configuration and installation of each individual security implementation.

It would be safe to say that foremost to any security implementation would be authentication services (of the source and the receiver[s]). Users must be confident with

whom they are communicating before communications are initiated. Once the identity of the networked participants is assured, data integrity (untampered data) services become essential. And lastly, confidentiality of message content rounds out the basic security requirement suite for typical networks. Methods of supplying these services will be discussed, and it should become clear that network purpose and design requirements will create unique sets of implementation challenges.

B. KEY DISTRIBUTION/MANAGEMENT (CHAPTER IV)

Chapter IV will discuss methods of key generation, transfer, verification, usage, update, storage, backup, and destruction (i.e., the multi-function umbrella referred to as key management). Simply stated, keys are shared secrets between consenting communications nodes. Their purpose is to prove possession of knowledge that identifies them as an approved communication entity. This is the most difficult task to accomplish with ultimate confidence in open environments. The user interface can thwart the best algorithm and/or protocol with a weak key that provides a security perception where little actual security protection exists.

Keys can be one of two types: secret (symmetric), or public (asymmetric). The majority of past cryptologic

implementations have employed secret key algorithms. Symmetric security mechanisms have been the historic mainstay. They use the same key to encrypt plain text and de-crypt cipher text, as shown in Figure 2.

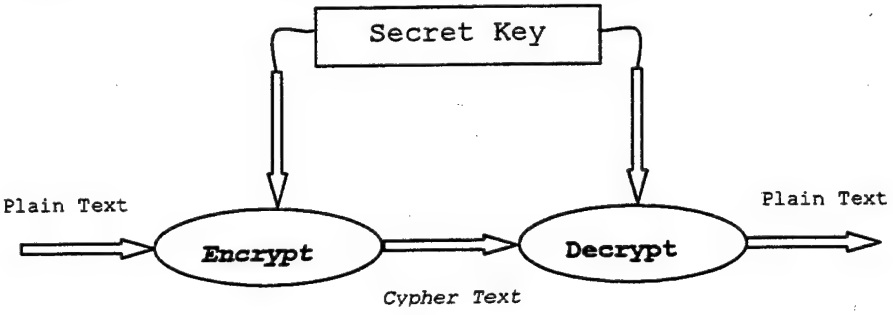


Figure 2. Secret (symmetric) Key Cryptography

Public key systems reflect a new security technology that uses a key pair (public and private) for separate encryption and decryption of data exchanges, as shown in Figure 3. Because public key methods are more computationally intense, they have been typically used to transmit common symmetric key material.

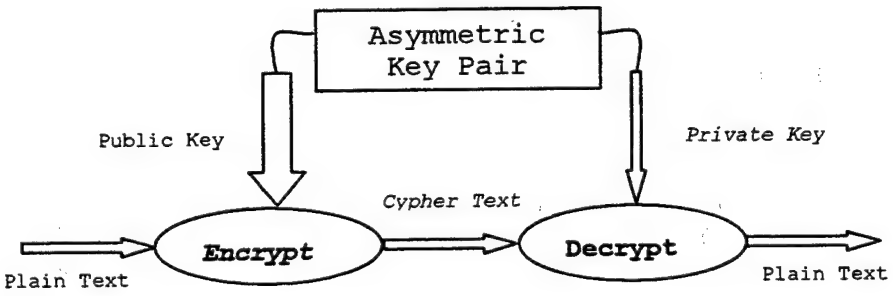


Figure 3. Public (asymmetric) Key Cryptography (data communications)

Key management in high security implementations make use of key management servers. These servers are called trusted third party (TTP) key distribution centers (KDCs for secret key networks) and certificate authorities (CA, the public key equivalent of KDCs) (Kaufman, 1995). The subject of TTPs rears the quintessential key distribution question, "whom can one ultimately trust?"

C. NETWORK (MULTICAST) SECURITY IMPLEMENTATION (CHAPTER V)

The last topic will combine the two issues previously mentioned above, and suggest security service considerations when planning secure tactical and naval data network protocols. Network applications (interactive vs. store-and-forward), performance requirements, mobility, user configuration dynamics, and node processing power all factor into the final network solution. TDNs can be cabled and/or wireless. Each network/segment may naturally support different security mechanisms with differing levels of effectiveness. Awareness of the features/challenges each available security technique offers will provide the focus for the discussion.

III. ESSENTIAL (NETWORK) SECURITY SERVICES (PRIMAL PRIMITIVES)

This chapter discusses essential security elements and methodologies required for multicast networks to operate with confidence that the information, which traverse these networks is not suspect, or compromised. Security requirements differ with each group of network users, and a one-size-fits-all solution will not be the optimum solution for any one networking purpose. Internetwork Open Systems Interconnection (OSI) characteristics challenges security systems to provide adequate protections, while allowing brisk data interchange among disparate networks/users. The ensuing discussion will focus upon core security services (authentication, integrity and confidentiality) and how they can be provided in network security protocols.

Section A discusses algorithms used to provide authentication and integrity services for network communications exchanges. Authentication services verify the identity of sender(s) and receiver(s), and integrity services provide assurance that the received data is the same as the data transmitted (i.e., detection of tampered data). Section B completes the algorithm discussion by addressing mechanisms that provide confidentiality to data exchanges. Confidentiality ensures that only authorized

parties are able to understand the contents of protected communication exchanges. Section C introduces security (e.g., authentication) protocols and how algorithms are employed to construct secure procedures (i.e., protocols) for protected exchange of data. Authentication protocols have been used in the discussion as a fundamental example illustrating the generic development of security protocols.

A. AUTHENTICATION AND INTEGRITY ALGORITHMS

A natural starting point for a discussion of multicast network security is authentication services. There are two distinctive types of authentication; data origin (message originator) and entity (the user, or system host). Security is difficult, at best, if one can not be sure who is sending data to the receiver and to whom the receiver is responding. A security service closely bound to authentication is data integrity (i.e., protection against unauthorized modification, deletion, or substitution since data creation, transmission, or storage). In fact, some sources consider data authentication and data integrity services like Siamese twins, inextricably joined together, one implicit in the other (Menezes, et al, 1996). In consideration of this viewpoint, discussion will include concurrent reference to both these services, and an overall algorithm association is shown in Figure 4. Every day examples of these services are

photo-identification cards (drivers License), watermarks, knowledge of one's social security number, or mother's maiden name (authentication), indelible ink, credit card hologram, or Notary seal (integrity).

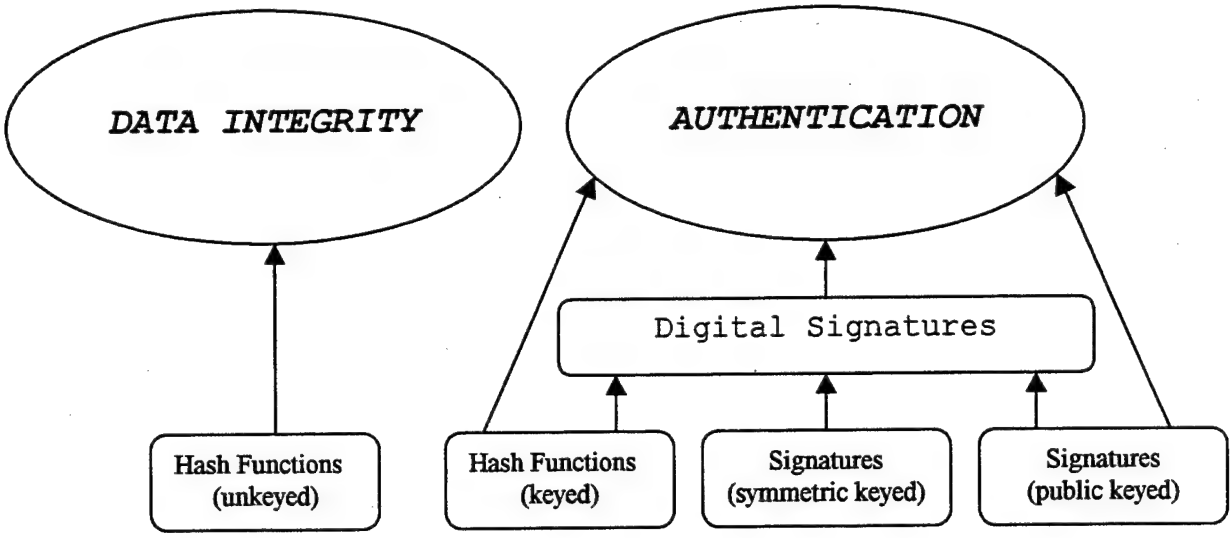


Figure 4. Authentication/Integrity Taxonomy. After (Menezes, et al, 1996)

Attacks related to authentication and integrity services can range from masquerade (an attacker pretending to be an authorized net member), replay (using legitimate recorded security exchanges to gain unauthorized entry), "man-in-the-middle" (monitoring all exchanges between targeted users), to mention the most obvious. OSI internetwork environments make these attacks more probable as new technology transforms the Information Super Highway into a global autobahn (an information cornucopia). Open internetworks have become a part of everyday business

(private, commercial, and governmental), but unfortunately the very feature that has been fundamental to its success, is now its nemesis. Like all things in life, payment is required in return for services (i.e., the cost to provide information security). That price can be increased processing demands, latency (message transmission time), message overhead, bandwidth, and protocol/algorithm complexity in varying amounts. When security implementations are planned, consideration must be given to the costs incurred by each mechanism.

Hash (one-way) functions and digital signatures are the basic security primitives used to provide authentication services, via hash (keyed) functions and digital signatures (symmetric and public key); and data integrity services, via hash (unkeyed) functions. Hashing functions are algorithms that take an arbitrary length input data message (M) and produce a fixed length (compressed) output (hash value, h). Hashing processes although more complex and secure are similar to parity functions used to validate data during transmission. The hash value (h) is a unique digest (fingerprint) of the input message (M). One-way hash functions (OWHF) have the following properties:

- given a message (M), the hash value (h) is easy to compute;

- given the hash value (h), it is difficult to determine the corresponding message (M); and

- given a message (M), it is difficult to find another message (M1) that produces the same hash value (h) as message (M) (strong, or collision resistant).

The uniqueness of these three properties provides a robust hash function, which underlies the resulting strength of the security primitive (and ultimately the strength of the security service). Keyed hash functions (e.g. message authentication codes [MACs]) are commonly used authentication primitives and unkeyed hash functions (modification detection codes [MDCs], or message integrity codes [MICs]) are basic integrity primitives. The MIC (an unkeyed hash function) operates on the message (M) to produce a unique digest of the message, this hash (digest) value is protected for later use and sent along with the message to the intended receiver. The receiver checks the integrity of a received message by computing the hash of the received message and comparing it to the protected hash value. Customarily cryptographic digital signatures are utilized to ensure the integrity of hashed values.

Few MAC algorithms (keyed hash functions) have been designed presumably because the algorithms originally developed have, so far, provided acceptable authentication

services. The MAC (a keyed hash function) operates on the message (an n-bit block) together with a symmetric (secret) key to produce the hash value (protected). The main premise is that the hash value can not be replicated without knowledge of the key, and knowledge of the key is proof of authenticity. A large number of these algorithms are based upon block-ciphers. There are hashing schemes where the block length (b) and the bit-length of the hash value (h) are equal.

Complex hashing algorithms can also produce hash values

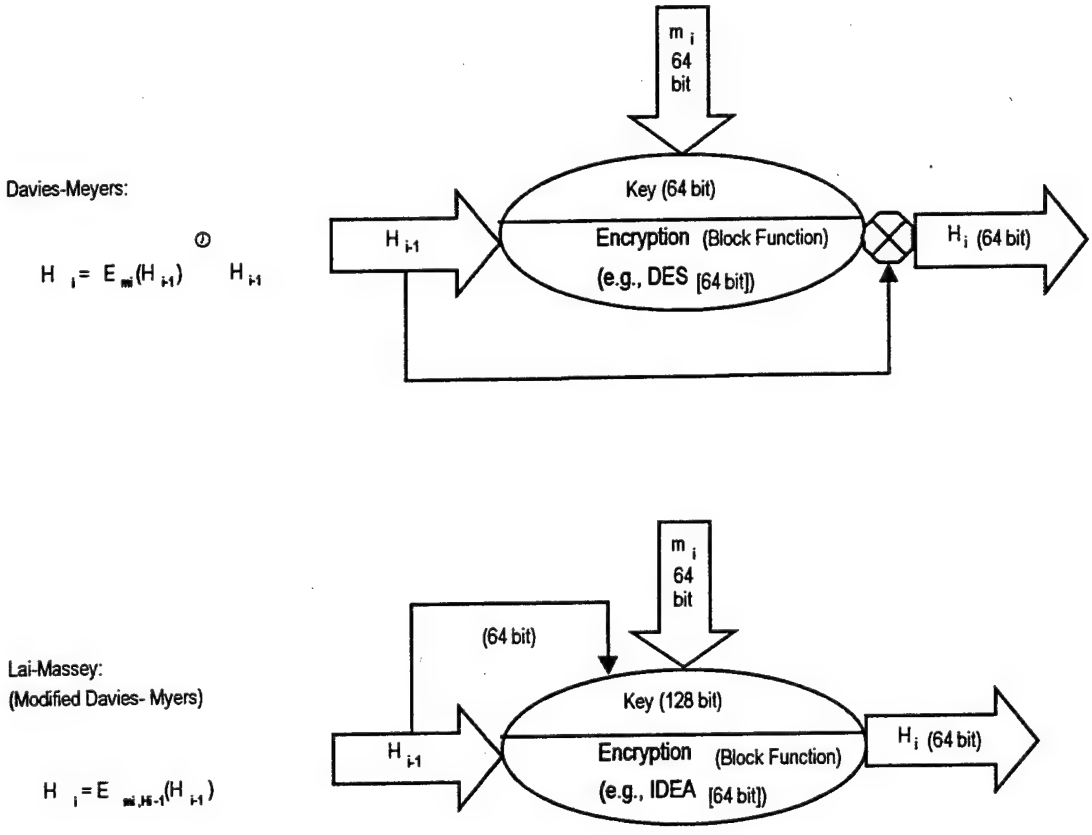


Figure 5. Hash Functions. After Schneier, 1996)

that are the same length as the message segments (i.e., block length). A modified Davies-Myer technique (shown in Figure 5) uses the International Data Encryption Algorithm (IDEA) with a 128-bit key on 64 bit message blocks to produce a 64-bit hash output value. Instead of combining (exclusive OR) the previous hash value (Hi-1) with the encryption of the previous hash value(E[Hi-1]) to form the current hash value (H, as in the original Davies-Myers technique), Lai and Massey form a current hash value (Hi) from the encryption of the previous hash value (Hi-1), using the previous hash value (Hi-1) with the current message block (mi) segment as the encryption key (vice just the current message block segment as the encryption key), as shown in Figure 5. Brute force (trying all possible concatenated key values) is the only way known to beat this hashing algorithm (Schneier, 1996). Present hardware computing capabilities are putting mainframe like computing power into a desktop work station, and the advent of distributed computing has increased the amount of processing power that can be accessed at a physical location, or network node. These performance gains sparked by hardware advances transforms the desire for longer (160-bit) hash values into a real requirement.

Two variations of Davies-Meyer hashing functions provide the added strength of a 128-bit hash value from 64 bit message block segments and a block encryption algorithm (like IDEA) with a 128-bit key. The Tandem Davies-Myer hash function (shown in Figure 6) uses two 64-bit key dependent encryption algorithms (sub-functions). Each encryption

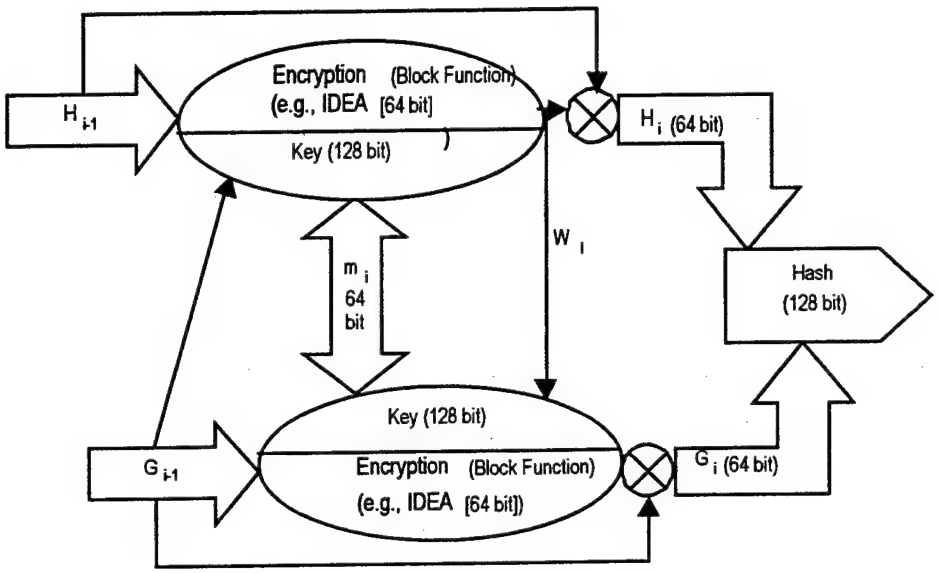


Figure 6. Tandem Davies-Myers Hash Function. After (Schneier, 1996)

algorithm receives a previously hashed 64-bit input and employs specially formed keys. One key is concatenated from the current 64-bit message block segment with the 64 bit hash input of one sub-function, and the other key concatenates the 64-bit hash output of the other sub-function with the current 64-bit message block segment.

These two sub-functions each produce a 64-bit hash value, which are used to obtain the final 128 bit hash value. This hash function is called tandem because the key of one sub-function is dependent upon the output of the other sub-function.

The other Davies-Meyer variation is called the abreast Davies-Meyer scheme (shown in Figure 7) because the keys are formed from a concatenation of the message block segment

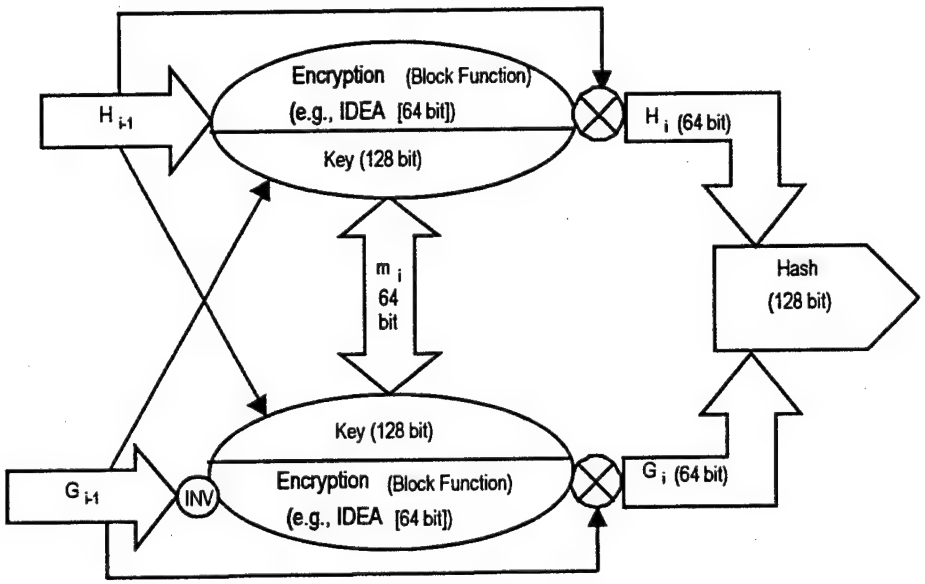


Figure 7. Abreast Davies-Myers Hash Function. After (Schneier, 1996)

with the hashed input of one sub-function, and a concatenation of the hashed input of the other sub-function with the message block segment. Neither key is dependent upon an output of either sub-function. Both these Davies-

Meyer hashing schemes yield secure (by today's computing standards) 128 bit hash values.

MDC-2 and MDC-4 are IBM proprietary hash functions and use the U.S. data encryption standard (DES) as the block cipher to form the output hash value. They also produce relatively strong hash values twice the length of the message blocks (Schneier, 1996).

Digital signatures are digest primitives that facilitate authentication services, by attaching a data string (unique to a particular sender/entity) to a message. Digital signatures can utilize public-key (asymmetric), or secret-key (symmetric) encryption and/or hashing techniques. The concept of digital signatures (using public keys) was introduced by Diffie and Hellman in 1976, but no practical implementation was developed until 1978 as put forth in the paper by Rivest, Shamir and Aldeman describing the concept of public key cryptography. Their implementation has become a popular standard known as RSA. Early (RSA) digital signature techniques provided message recovery, subsequent digital signature methods (with appendix) employ hash functions to construct the digital signature from a hash of the original message. These digital signature methods (with appendix) require the original message be input into the verification function.

Examples of digital signature schemes (with message recovery) are asymmetric (public-key) cryptographic schemes by RSA, Rabin and Nyberg-Rueppel. It should be noted that any asymmetric (public-key) cryptographic digital signature scheme (with message recovery) can be converted into a digital signature scheme (with appendix) by hashing the message and signing the hashed result (Menezes, et al, 1996).

A complete (end-to-end) digital signature process requires three basic mechanisms; a public/private key pair generation scheme, the digital signature generation scheme and a signature verification scheme. Recall that digital signature processes can be described as one of two types; digital signatures with appendix (requiring the original message as input into the verification algorithm), or digital signatures with recovery (the original message is not needed for verification, but is obtained from the signature).

Examples of digital signature schemes (with appendix) are (Menezes, et al, 1996):

* ElGamal which uses a random number (a generator) to compute a public/private key pair and a hash (or, redundancy) function to form the signature with which to sign a message of arbitrary length,
* DSA (Digital Signature Algorithm) proposed in 1991 by the U.S. National Institute of Standards [NIST] as U.S.

Federal Information Processing Standard [FIPS] 186 Digital Signature Standard [DSS] is a variation of the ElGamal scheme and

* Schnorr signature algorithm (another variant of the ElGamal scheme). Variants of the ElGamal technique have been modified with private (symmetric) encryption algorithms to produce digital signatures with message recovery.

Techniques for one-time signatures (i.e., each message requires its own public/private key pair, e.g., Rabin, Merkle, and Goldwasser/Micali/Rivest [GMR]) are very efficient algorithms though not practical (Menezes, et al, 1996). One-time signatures can be used for multiple messages when combined with authentication trees, but additional work is required to construct the tree and calculate intermediate node parameters (hash values) required by this authentication tree scheme.

Authentication trees are binary trees that contain a node for each message requiring a digital signature. A message (corresponding to a node positioned below the root node) has that node's signature verified through a series of comparisons of hashed public parameters and signature parameters associated at each node along the shortest path (the authentication path) between the node (of interest) and the root node. The public and signature values of the root node are determined by a trusted third party (TTP). All

these values are used to provide the authentication for a message associated with a particular node of the authentication tree. The lower down the tree the node (message) is located, the more public/signature parameters must be provided by the sender and verified by the receiver to ensure message signature authenticity (Menezes, et al, 1996).

After authentication issues have been addressed and network users are sure (security is just a relative level of immunity to attack judged as practicable, sufficiently difficult, computationally prohibitive, intractable, etc.) of identities/signatures, the security spotlight falls upon the issue of data integrity. If an adversary can stir up the data stream so that the data is rendered unusable, then both sender and receiver are deprived of network services (denial of service). More serious is the condition where an adversary can modify data without such modifications being detected. These modifications could be random bits throughout the data stream (disruption), or more seriously, the modifications could be intelligent changes to the data made at the whim of the intruder (a major security failure). MDC/MIC/MD (unkeyed hash) functions have been mentioned as the primary primitives that are applied to data integrity issues. These functions are usually one-way

(difficult/impracticable/intractable, etc., to reverse engineer), take the message as input, and return a fixed length string unique to the input message, i.e., collision resistant (computationally difficult to find two inputs that yield the same output value). These functions typically employ block ciphers, modular mathematics, or custom hashing schemes designed for fast execution. Examples of block cipher hashing schemes are: Matyas-Myer-Oseas, Davies-Myers, and Miyaguchi-Preneel (the preceding are single-length, hash output length[n] is equal to the encryption block length, e.g., for DES, n = 64 bits), MDC-2 and MDC-4 (MDC-2 and MDC-4 are double-length, n is twice the length of the encryption block, 128 bits, because both MDC-2 and MDC-4 use the 64 bit DES scheme). Examples of customized hash (message digest [MD]) functions are MD-4, MD-5, secure hash algorithm (SHA) 1 (U.S. FIPS 140-1), and RIPEMD-160 (a result from the European RACE Integrity Primitives Evaluation project which produces a 160-bit hash value). The strength of SHA-1 and RIPEMD-160 have been judged equal, they appear to offer the best protection, but are four times slower than MD-4. Hash functions employing modular arithmetic (from public key systems) have tended not to possess the strength of the custom hash functions discussed above. For this reason, little interest has been shown for their practical use

(Menezes, et al, 1996). Taken together, authentication and integrity services provide the cornerstone for providing privacy in an open, inter-connected, public world. The last fundamental element of security, confidentiality, is needed to assure sender and receiver(s) that they are free to communicate securely in an open environment.

B. CONFIDENTIALITY ALGORITHMS

Encryption methods are the means by which a data stream is made confidential (i.e., safe) so that only cooperating parties (and not adversaries) can share and make use of information within the data stream. It was mentioned in the introduction (Chapter I) that encryption is not a new requirement, but has spanned the centuries. Early uses of encryption guarded the secrets of state, then the affairs of business, and today average civilian user finds these same capabilities a requisite part of their personal daily life. Stream ciphers, secret-key block ciphers, and public-key encryption are the three most commonly used techniques that ensure data confidentiality, Figure 8 shows this association.

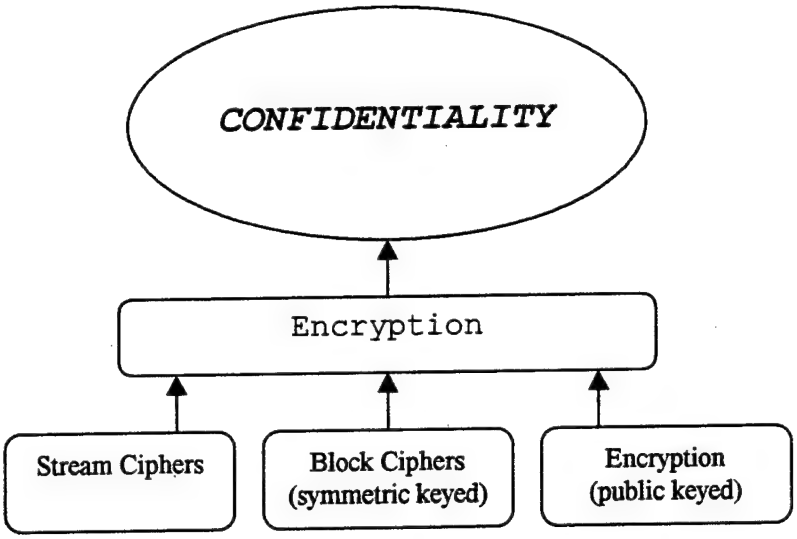


Figure 8. Confidentiality Taxonomy. After (Menezes, et al, 1996)

Stream ciphers operate on single characters (small block sizes), or bits and are capable of higher throughputs than other encryption techniques. They are well suited to hardware implementation, software implementations of stream ciphers tend to suffer performance reductions. There have been recent, increasingly efficient software implementations proposed, but they are proprietary and can support 2 - 7 Megabits per second throughput (Menezes, et al, 1996). Stream ciphers commonly use feed back shift registers (linear and nonlinear), and can require sender/receiver(s) to use synchronized key streams, or they can be designed to operate as self-synchronizing cipher streams. These types of ciphers are especially adaptable to wireless communications applications which have intrinsically noisier

(or hostile radio frequency [RF] environments) links/transmission media. The use of specialized hardware to implement these stream ciphers would tend to reduce the openness of the networks that they operate in. With the cost of hardware spiraling ever-lower year by year, these costs could become insignificant and the real issue would be the transparency with which they could be made to operate in open networks.

Block ciphers are a staple of the cryptographer. These are the atomic primitives that many other primitives (e.g., pseudorandom number generators, stream ciphers, MACs and hash functions) are made of. These ciphers can be symmetrically (secret), or asymmetrically (public) keyed, and usually operate on 64-bit (or larger) message blocks. The oldest forms of these ciphers are the secret keyed block ciphers. They have provided effective service to Roman legions and have flourished through the centuries to present day. Secret keyed ciphers rely on the strength of the key to provide message confidentiality. The earliest of these schemes were transposition, or substitution ciphers and have progressed as variations of these elementary techniques. Transposition reorders the plain text block (unencrypted text string) through the operation of a permutation function e.g., the cipher text block "asarec" is decrypted as

"caesar", when the permutation function (e = 542631) is applied to the text block. Substitution ciphers replace each character (or character group) of plain text with a predetermined character (or character group) to form the cipher text. The cipher text can be from the same alphabet (mono-alphabetic) as the plain text, e.g., the following cipher text block "f d h v d u" can be decrypted as "Caesar", when the substitution function employs the same alphabet and is a simple three character shift to the right (the ciphered character (value) minus 3 mod26, when the letters a - z are given numeric values 0 - 25). The substitution symbol set used to encrypt the plain text can be changed for each text character (polyalphabetic substitution) of the message on a predetermined schedule. Rotor machines implement polyalphabetic substitution with multiple 26 character wheels (the famous German Enigma rotor machine had three) that are stepped (offset) as each proceeding wheel is moved. The resulting encryption features these machines provided were long intervals between alphabet set repetition and large offsets between alphabet changes.

Block ciphers operate on fixed bit length segments (blocks) by breaking the message (M) into a multiple number (i) of fixed-bit-length(b) message segments (i.e., blocks)

to produce compressed hash values (e.g., MDs). Message padding is used to ensure that the input messages are always a multiple of the selected block length (b).

Cipher-block-chaining (CBC) is a feedback block cipher scheme that commonly uses symmetric ciphers to iteratively compute a hash value for each message block segment (i). Symmetric (secret key) ciphers use the same key for encrypting clear text into cipher text, and decrypting cipher text into plain message text. The block cipher process starts with an initial value (initialization vector [IV]), combined (via an exclusive OR operation) with the first block (message segment [m1]). The IV is the same bit length(b) as a message block (and sometimes contains all zeros). The ORed result (2b bits) is fed into the encryption algorithm with the encryption key (k) to produce an encrypted output (e1). The encrypted output (e1) is combined (i.e., ORed) with the next message block (m2) and inserted (with the key) into the encryption algorithm to produce another encrypted output (e2). The recent encrypted output (e2) is combined with the next message block segment (m3) and run through the encryption algorithm, producing e3. This process repeats i (the number of message block segments) times, the last encrypted output (ei) is used to

produce a final hash value (h) which (in this example) is twice the length of the block size (b).

An alternate version of the above scheme uses the message block (b) as the key (k) and the preceding encrypted output(e) as input (vice the message block) into the encryption algorithm. The advantage is that a succession of differing blocks (message segments) are used as keys for the encryption (vice using the same key repeatedly). Since most block cipher algorithms use block lengths of 64 bits, hashed output values of 64 and 128 bits are common. The method described above is a simplified description of block cipher techniques employed in practice.

Today, DES (data encryption standard, FIPS 46-2) is the most renowned encryption technique throughout the world. This cipher operates on 64 bit (message) blocks with a 64 bit key (but only 54 bits are used in the computations) by performing substitutions and permutations (SP rounds) iteratively (16 SP rounds) on 32 bit combinations of the intermediate encryption results. A final combination of the 32 bit halves produces the final 64-bit cipher block output. The key length (56 bit) controversy has always clouded discussions on the strength of this encryption mechanism.

IDEA (international data encryption algorithm) uses a 128-bit key and operates on 64 bit message blocks to produce

64 bit cipher blocks. It operates in a similar manner to DES, splitting the plain text input (64 bit) block into four (vice two, as in DES) 16 bit sub-blocks for eight rounds of processing, and then combining the 16 bit results into a 64 bit cipher output block. This algorithm is considered by some cryptologists as the strongest publicly available block cipher today (Schneier, 1996).

RC5 is a word-oriented block cipher that was designed for efficient software, or hardware implementation. The word lengths (w) can be variable (16, 32, 64 bits), but are generally 32 bits, which results in a block length (twice the word length) of 64 bits. Much analysis has been performed on the 64 bit data block version (of RC5) and its strength appears promising for applications requiring more efficient software implementations (Schneier, 1996).

SAFER K-64 (secure and fast encryption routine, with 64-bit key) is another iterated block encryption algorithm that takes 64-bit plain text blocks and produces 64-bit cipher text blocks. A minimum of six rounds of iteration has been recommended for this algorithm, ten rounds are the maximum.

Khufu and Khafre are similar to DES and have been intended as alternative fast software implementations to the DES cipher. Khufu can use up to 512 bit keys and Khafre can

use key lengths that are a multiple of 64 bits. Both schemes can employ various rounds of iteration. Khufu (16 rounds) and Khafre (4 rounds) have been found to be more resistant to the best currently known cryptographic attacks than DES (Menezes, et al, 1996). There are other block ciphers, and descriptions of them can be found among the references in the reference list. The second major group of encryption systems is the asymmetric (public key) schemes.

The most celebrated public encryption method is the RSA scheme, (named for its developers Rivest, Shamir, and Adleman), it can provide data confidentiality and authenticity (via digital signatures) services. The essence of the strength of the RSA algorithm is the adversity in reducing a large integer into a product of prime factors raised to integer powers. The uniqueness of these encryption systems is the public/private key pairs that allow encryption/decryption, respectively. Public key ciphers are slower than their private key counterparts, but public key methods can reduce the number of keys required for operation on networks. The basic RSA system generates a key pair (e,d), uses the public key(e,n) to encrypt and the private key(d,n))to decrypt message blocks(m), where n is the product of two large random prime numbers (p & q). The product (n) is the modulus used in the encryption and

decryption algorithms. RSA implemented in hardware has been shown to be 1000 times slower than DES ciphers, and software RSA implementations are about 100 times slower than DES software implementations (Schneier, 1996). This relative speed disadvantage has relegated RSA ciphers for use in protecting symmetric session keys and digitally signing small document exchanges. The security of RSA has stood the test of time, and is optimized when: the modulus (n) is not shared with other encryption/decryption key pairs (e,d); the decryption key (d) is large; and the message blocks are appended with random values (salted) to strengthen the use of small encryption keys (e). Moduli (n) lengths of at least 768 bits are recommended for security in today's current hardware technology environment, and a modulus length of 1024 bits should be used if keys are to be securely used over a long term (Menezes, et al, 1996).

Other methods of public key encryption are Rabin, ElGamal, and the Chor-Rivest knapsack scheme. Rabin and ElGamal employ modular mathematics to compute the public/private key pairs and encrypt message blocks and to decrypt cipher text blocks. Rabin encryption is faster than RSA and the decryption algorithm is of comparable speed (Menezes, et al, 1996). Rabin public key encryption was the first provably secure public key scheme. ElGamal employs

similar logarithmic constructs to those used in the Diffie-Hellman key exchange algorithm. The Chor-Rivest knapsack algorithm is one of the few secure knapsack public encryption methods. The knapsack technique is based upon a subset of sums in which a simply determined (superincreasing) subset sum (corresponding to a message block sequence of 1's and 0's) is concealed to appear as an instance of a larger (and more difficult to solve) subset sum problem (Schneier, 1996).

RSA, Rabin, and knapsack methods are examples of deterministic encryption schemes. Deterministic encryption schemes yield the same cipher text (c) for a message (m), and it is easy to detect when message transmissions have been repeated. A randomized constant length bit string can be added to the message blocks to produce cipher text that would not repeat, but the resulting scheme would not be provably secure against all conceivable attacks.

Probabilistic encryption is a provably secure form of public encryption that combines random values with the message blocks during the encryption process. The Blun-Goldwasser scheme is an example of probabilistic public key encryption. This is the most computationally efficient probabilistic type algorithm, and is very comparable to RSA

in terms of speed and message-to-cipher block size increase (Menezes, et al, 1996).

The algorithms used for authentication, integrity, or confidentiality all have weaknesses, and differing levels of overhead that burden the communications channel. The methods discussed are considered to be some of the most efficient, effective and recognized. Nuances are continually being made to existing algorithms with the intent to provide stronger and more efficient security services.

The manner in which the algorithms are employed (the protocol) can mitigate some of the shortcomings that an algorithm may exhibit. The construction of basic primitives (algorithms) is specialized and best left in the hands of experts, much like specialized software drivers are developed by those with the required esoteric knowledge. The Security protocol designer will incorporate those security primitives that best fit the design conditions specified by network operation requirements.

C. AUTHENTICATION PROTOCOLS

Basic authentication protocols prove the sender's(S) identity to the receiver(R) by having the sender perform a cryptographic operation on an artifact supplied by the receiver. This only authenticates the sender to the

receiver (one-way authentication). For both parties to be sure whom they are communicating with (mutual authentication), requires the receiver to perform a cryptographic operation on an artifact supplied by the sender. This can double the number of exchanges between sender and receiver. Well-designed protocols achieve mutual authentication with fewer exchanges.

A simple example (shown in Figure 9) using public key

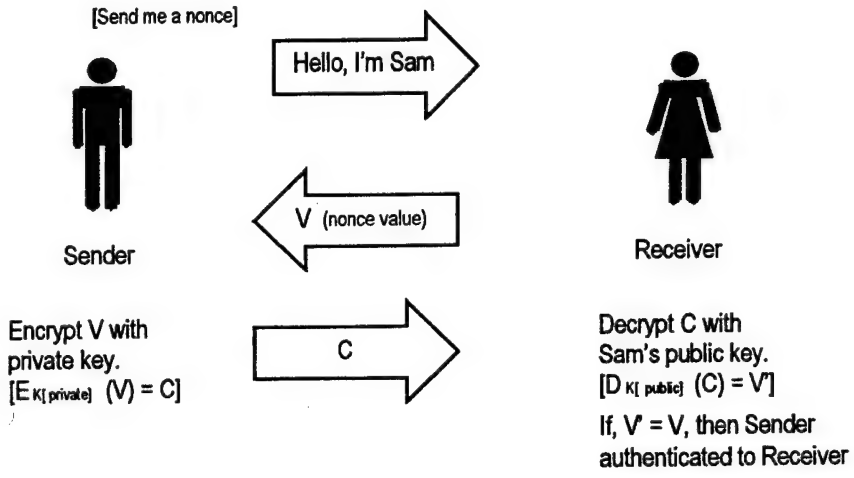


Figure 9. Basic Public-Key Authentication (one-way)

cryptography consists of sender (S) identifying himself to receiver (R). The receiver sends a value(V) to the sender, the sender encrypts the value(V) with his private key and sends the result to the receiver, the receiver uses the sender's public key to recover (decrypt) V, if the recovered

value matches the transmitted value, then the receiver verifies the sender's identity.

The exchange described above is one example an authentication mechanism that can be applied in a distributed (networked) system of computers (nodes). Typically, passwords (identifying an individual, or a host machine) and/or host (machine/node/IP) addresses have been used to provide authentication services to users. User passwords routinely authenticate an individual to a node (host computer), and the host IP (node) address is used in authentication of the user node to other nodes on local, intra, or inter-networks. The focus of this discussion is upon authentication within a distributed system of nodes.

A brief word on inferred user-to-host network authentication is necessary. In the early era of networking, when private, dedicated (i.e., stovepiped) communications channels were established, network users could be relatively secure in the belief that only friendly, authorized users had access to the network. Today, with the advent of inter-connected systems of networks (i.e., the Internet) that belief has totally been supplanted with a zeal to protect the host network from the threat of outside nefarious activities. This zeal is confirmed by the interest in, and growing supply of network firewalls,

devices to filter/sanitize incoming (and outgoing) network message traffic, and Internet filters (like GuardiaNet ® [server based], or Cyber Patrol® [client based]). These precautions require additional devices/processes to be operative, increase operation/maintenance costs and induce performance burdens to distributed hosts/systems (both administrative and tactical networks). It should be clear that authentication services may no longer be thought of as add-on accessories to network operations, but instead operations designed and integrated into the network protocol.

Security services are major features, along with a badly needed increase in IP address space (to 128-bits) of the new Internet protocol standard, IPv6. IPv6 provides authentication and integrity services through the incorporation of an authentication header (AH) and confidentiality/integrity services are provided by an encapsulated security payload (ESP).

The IPv6 authentication calculation is generally performed using a message digest algorithm (e.g., MD5) and either encrypting the message digest, or keying the message digest directly. Only cryptographically strong one-way algorithms are recommended for use with the authentication header. When block-oriented authentication algorithms (e.g.

MD5, MD4) are used and the IPv6 packet is not an integral number of blocks, the authentication data calculation is performed using zero bytes to pad (at the end) the message length to an integral number of blocks. The sender computes the authentication over the packet as it will appear to the receiver. The computation allows for future IPv6 optional headers that the receiver might not know about, but are known to the sender, when such options are included in the packet. The sender places the encrypted output (calculated message digest) into the Authentication Header. When a packet containing an IPv6 Authentication Header is received, the receiver independently calculates the authentication header data for the received packet. The receiver compares the received Authentication header contents with the authentication header value it calculated. If the two quantities match, the received datagram is accepted as authentic (and unmodified), and when these quantities differ, the receiver discards the received datagram as non-authentic (and/or modified). The receiver audits authentication failures using conventional operating system facilities. This authentication process assumes that the strength of the message digest function and the management techniques are adequate for the network security environments traversed between sender and receiver.

Contrast IPv6 packet authentication with the most prevalent and convenient authentication method presently in use by most distributed systems today; host address-based authentication. Hubs may, in many cases, refuse to forward packets with incorrect data-link addresses, but incorrect (or spoofed) network layer addresses would generally be delivered unchecked. And even if routers were trusted, by adding authentication features to them, current IP source routing features provide a means for an interloper to specify the route packets travel from sender to receiver. This allows an interloper to be placed on a path between an unsuspecting sender and receiver, intercept authentic messages and inject false, or modified messages.

Authentication protocols should protect against eavesdroppers (passive attacks) and impersonators (active attacks). Passive attackers should not be able to obtain contents of messages between sender and receiver that would allow off-line password-guessing, or subsequent impersonation of sender/receiver. Active attackers should be prevented from hijacking a message exchange between a sender and a receiver, changing/rearranging/replaying message contents, reading a senders/receivers database (keys), initiating a conversation as the sender, accepting a connection as the legitimate receiver, deceiving

sender/receiver into decrypting, or signing chosen text (Kaufman, et al, 1995).

Review of a simple secret-key protocol implementation will help identify/describe typical message exchanges (five) required, between initiator (sender) and receiver (respondent) to achieve mutual authentication.

- a) The initiator (Igor) sends a message to the respondent

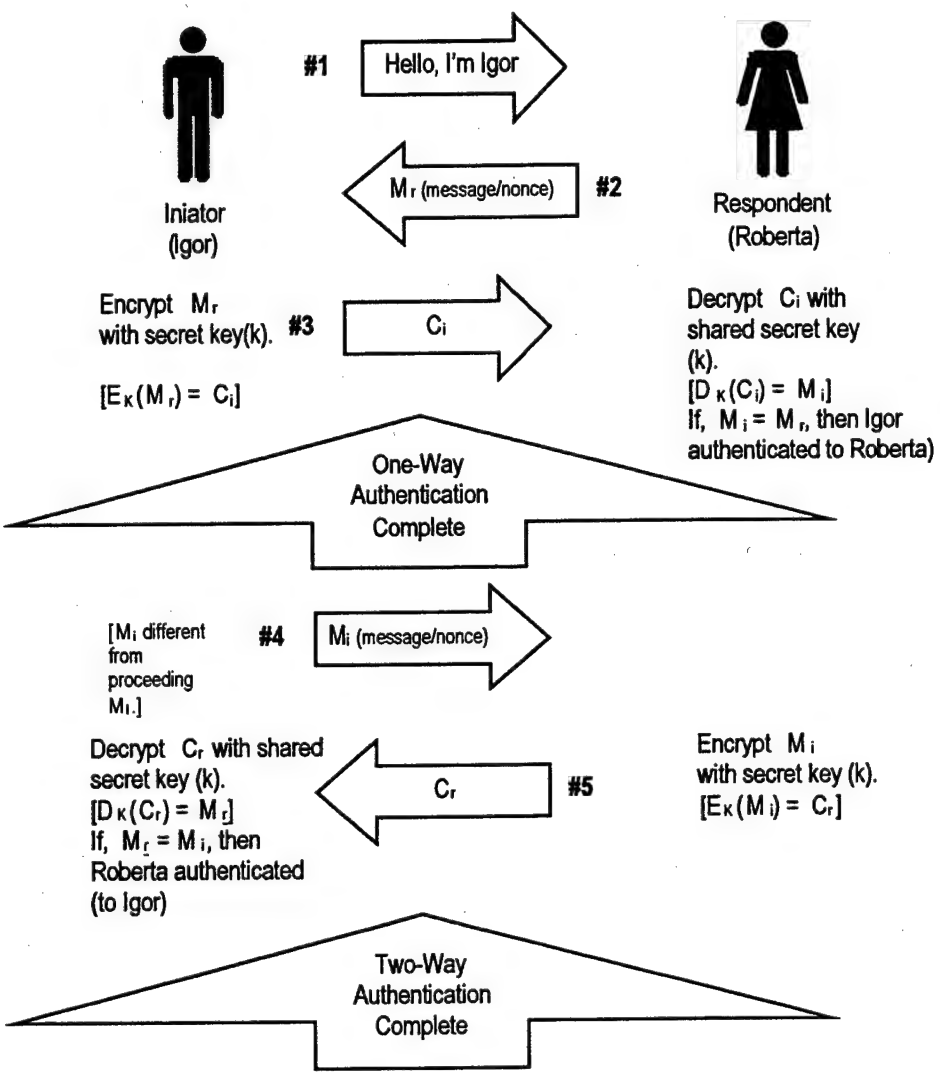


Figure 10. Authentication Protocol Exchanges (Secret-Key)

(Roberta), "I (Igor) want to speak with you".

- b) The respondent (Roberta) answers with a plain text message (Mr) to Igor.

- c) Igor encrypts [E(Mr)K = Ci] (or signs) Roberts's message (Mr) with the symmetric key (K, a shared secret between Igor and Roberta) and sends the result (Ci) to Roberta. Roberta uses the symmetric key (K) to decrypt the message from Igor [D(Ci)K = Mi]. Roberta then compares the decrypted message from Igor (Mi) with the message she sent to Igor (Mr), if they match (Mr = Mi), Igor's identity is authenticated to Roberta.

- d) Igor sends his message (Mi) to Roberta.

- e) Roberta encrypts [E(Mi)K = Cr] Igor's message and sends the encrypted result (Cr) to Igor. Igor uses the symmetric key (K) to decrypt the message from Roberta [(D(Cr)K = Mr]. Igor then compares the decrypted message from Roberta (Mr) with the message he sent to Roberta (Mi), if they match (Mi = Mr), Roberta's identity is authenticated to Igor.

Security primitives other than secret key encryption can be used to develop mutual authentication protocols, the example above is just one (of many) possibilities. The example protocol discussed is effective, but not efficient. A public key variant (shown in Figure 11) can perform the same mutual authentication task in three exchanges, as follows:

- a) Combine exchanges a & d (e.g., Igor declares his intention to speak with Roberta and sends a message (Mi) to Roberta.)

- b) Combine exchanges b & e (e.g., Roberta encrypts Igor's message (Mi) with her private key (Kr) and sends the result (Cr) and a message (Mr) to Igor.)

- c) Igor decrypts Cr with Roberta's public key (Krp), if the decrypted message (Mri) matches the original message (Mi) sent to Roberta, Roberta is authenticated to Igor. Igor encrypts the message from Roberta (Mr) with his private key (Ki) and sends the result (Ci) to Roberta. Roberta decrypts Ci with Igor's public key (Kip), if the decrypted message (Mir) matches the original message (Mr) sent to Igor, Igor is authenticated to Roberta.

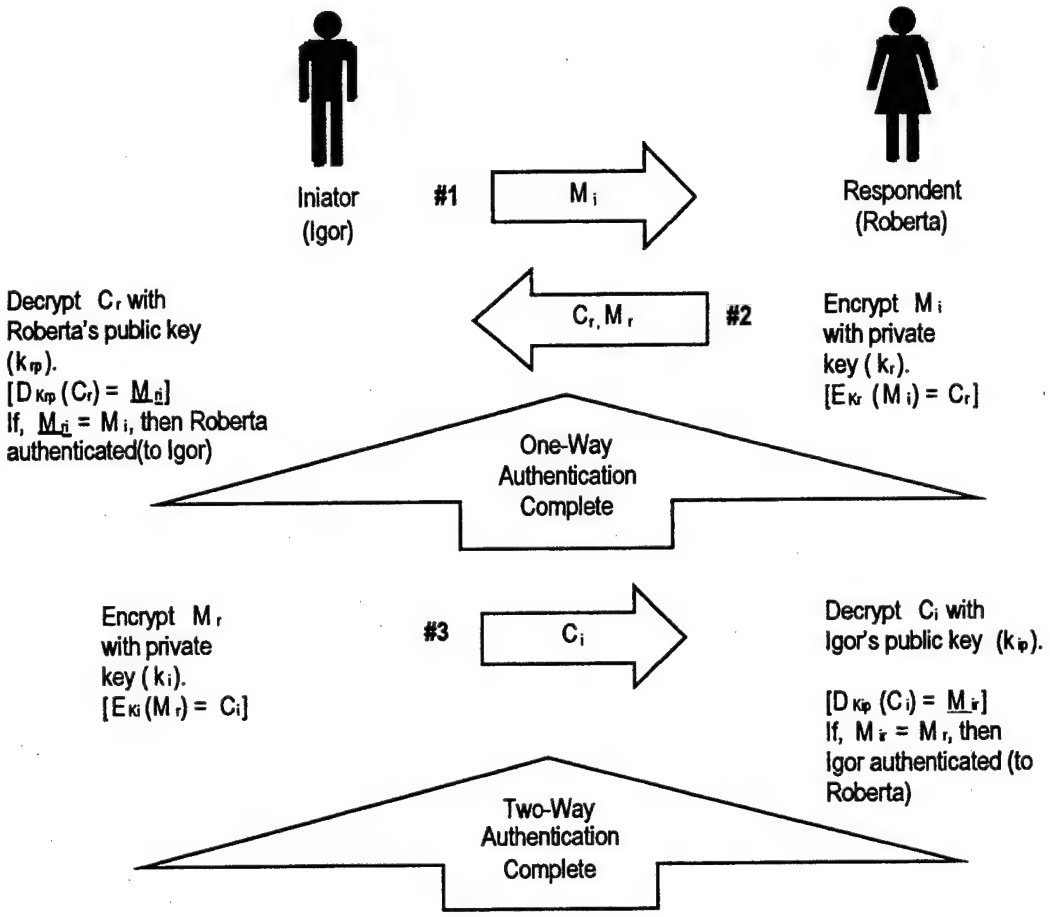


Figure 11. Authentication Protocol Exchanges (Public-Key) After (Kaufman, et al, 1995)

This same protocol could be accomplished with secret key primitives, but would be susceptible to reflection attack.

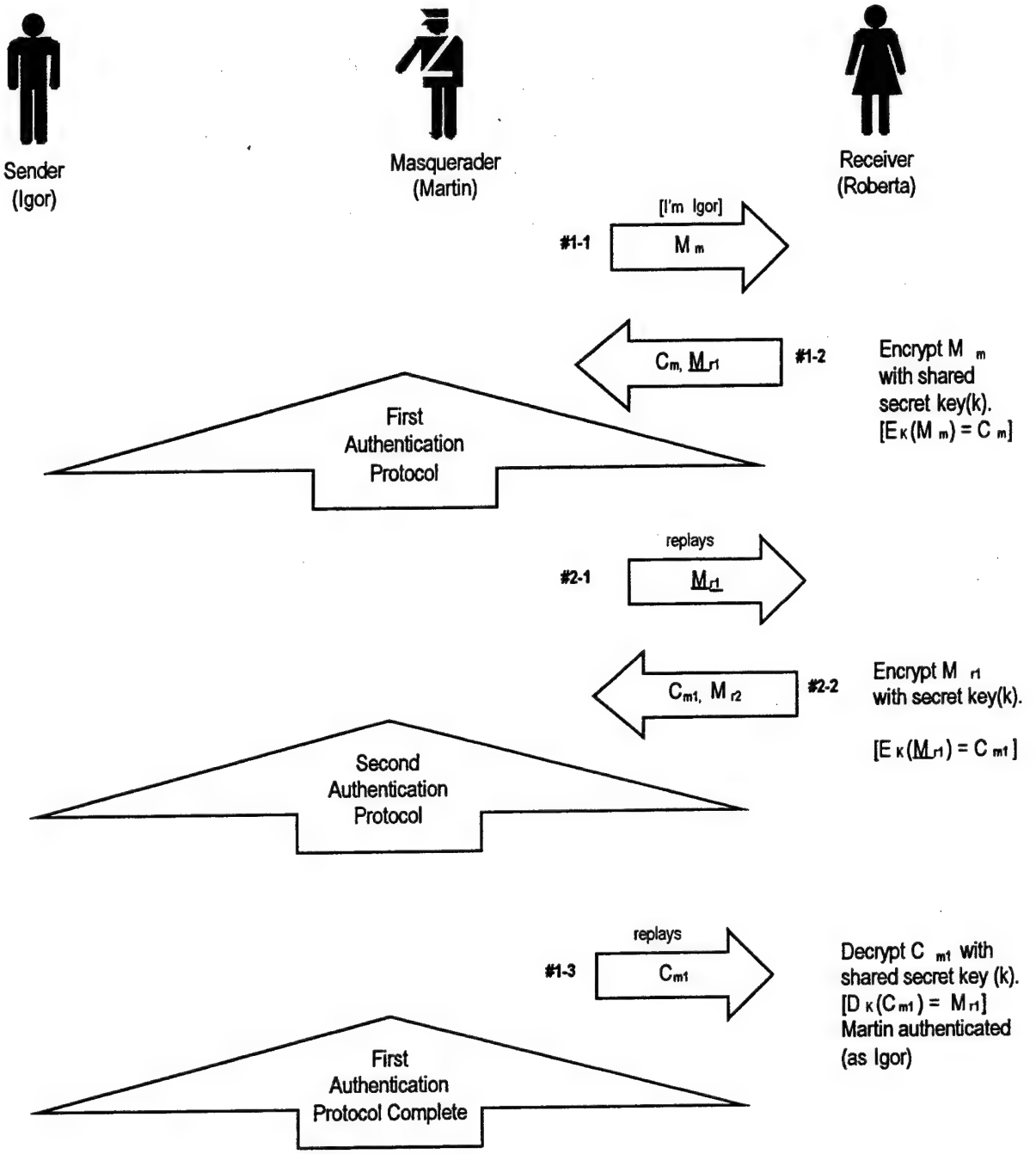


Figure 12. Masquerade (Reflection) Attack After (Kaufman, et al, 1995)

A reflection attack (shown in Figure 12) is an active encounter, by an individual (Martin) masquerading as a

legitimate party (e.g., Igor) of the communications exchange. The masquerader (Martin) initiates an exchange with Roberta, claiming to be Igor, but can not complete the first exchange, because Martin does not know the shared secret key (K) to encrypt Roberta's message (challenge, Mr). Roberta also returns Martin's encrypted challenge (nonce). Martin opens a second exchange with Roberta, supplying (replaying) Roberta's challenge (Mr), from the first exchange, as his challenge for the second exchange. When Martin receives the encrypted response (E[Mr]K) to this challenge, he uses this response to complete the first authentication exchange.

A public key version of the protocol described above does not suffer from reflection attack, but will have other impediments to be reckoned with. For instance, knowledge of the receiver's public key and the initiator's private key must be securely obtained at the network node the initiator intends to communicate from. Securely obtaining correct keys (in open netowrks) is a problem in key distribution and management, which will be addressed in the following chapter.

Authentication protocols can be even more efficient and require only two exchanges (one for Igor, and one for Roberta) to complete a mutual authentication exchange for a

communications session. This technique uses timestamps as the challenge from each of the communicating parties. The initiator (Igor) encrypts a timestamp (t) and sends the encrypted (E[t]=Ct) timestamp to the receiver. The receiver decrypts the ciphered timestamp (Ct) and checks that the result (t) is within a pre-established interval of time. Roberta (the receiver) increments the timestamp (t+1), encrypts the incremented timestamp (E[t+1]), and sends it back to Igor (the initiator). A similar check is made (by Igor) to ensure that the decrypted timestamp is within the acceptable time window. The major benefit of this technique is that it is easy to incorporate within existing request/response network protocols, but this protocol requires a means of providing time-synchronized information to the communicating parties, and care must be exercised in selecting an appropriate time window for use in authenticating legitimate participants **(Kaufman, et al, 1995)**.

Authentication protocols can only provide the level of security that has been designed into them. Networks and their associated security requirements can vary greatly, but key distribution and management functions are common elements of all security systems. The next chapter discusses common key distribution and management techniques

to highlight the viewpoint that key management mechanisms operate most efficiently when designed for specific network environments. Key management techniques must also provide scaleable features to accommodate multiple network users and topologies. These additional capabilities (multicast users, scalable topologies) can tax performance in simpler (less demanding) network security environments.

IV. KEY DISTRIBUTION/MANAGEMENT

Nearly all security primitives require that a special quantity (secret/public/synchronized/random, etc.) be utilized to facilitate the rapid recovery of intelligent information from a stream of seemingly random ones and zeros. This special value functions as a key that easily unlocks esoteric information from apparent gibberish. Security mechanisms protect information from disclosure for a predetermined time (likened to information half-life). As technological advances are made, methods of information protection must appropriately develop into stronger (more resistant) security techniques. For the better part of past cryptologic history, the cryptologic key has been the strength of the protection method. The longer the key length, generally the longer the time required (by an adversary) to perform a brute force (i.e., try all possible keys) attack against the security primitive. Longer keys in the past have equated to extended secure information life, but today stronger security practices (e.g., increased key length) are required to maintain secure information lifetimes.

Keys for security primitives must be delivered to only authorized workstations/hosts/users in public (open)

networks in ways that minimally impact network performance. This sounds like the search for the Holy Grail, but realistically is an exercise in engineering trade-off analysis to design and then implement the required level of information security consistent with desired data throughput. To effect secure message delivery, open, in-band key transfers need to be securely performed. There are several methods in use, which satisfy this goal.

Methods of key distribution can range from a clandestine meeting (an out-of-band-mechanism), using trusted servers (physically secured), to an open network exchange (Diffie-Hellman). The ensuing discussion will focus upon in-band open network distribution of keying material.

A. KEY ESTABLISHMENT

The requirements for key establishment (transport/agreement) over open networks differ for symmetric (secret) and asymmetric (public) keys. Distribution of public keys generally require only authentication and can be transmitted openly (unciphered), while secret keys must be confidentiality transmitted (ciphered) and authenticated.

Key hierarchy (shown in Figure 13) permits long-term keys (at the top of the hierarchy) to be used to securely

transport lower level interchange keys (e.g., secret user keys, session keys, etc.) (Ford, 1994).

Trusted/centralized servers (e.g., trusted third party [TTP], authentication servers [AS], key distribution centers

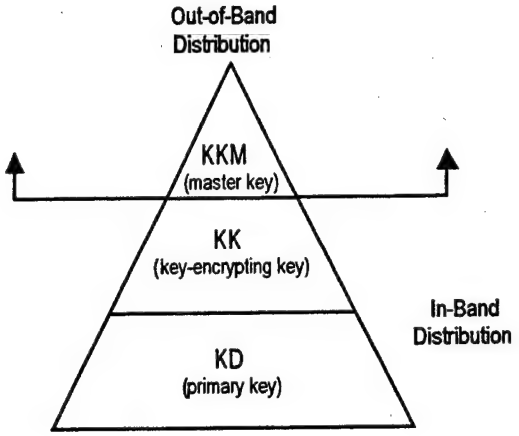


Figure 13. Key Hierarchy

[KDC], certification authorities [CA], and key translation centers [KTC]) provide communicating parties (unicast pairs, or multicast groups), with off-line key establishment services which may be employed to accommodate node (network)

Property	*KKM (level 1)*	*KK (level 2)*	*KD (level 3)*
Terminology	Key-encrypting Key Master (master key)	Key-encrypting Key (master key)	Key-encrypting Data (session key, primary key)
Duration	Long	Long	Short
Protection	KK & KD	KD	Data
Distribution	Out-of-Band	In-Band	In-Band

Table 1. Key Hierarchy

scalability requirements. On-line, trusted third party servers also provide real-time, mediated key transport

services. These mediated key establishment methods (e.g., Kerberos version IV and V) provide either one-way, or mutual authentication and secure key establishment to participating entities. Interactive mediated key distribution schemes have difficulty scaling up to accommodate large user node networks, because of central (key management) server processing limitations. On the plus side, central server based protocols can readily adapt to changing situations that may compromise user keys, or conditions that rescind user access to key materials.

Key establishment techniques either transport a key (determined exclusively by one party) between users, or allows two (or, multiple) parties to corroborate equally in developing (i.e., agree upon) a shared secret key (e.g., Diffie-Hellman key agreement). The objective of most key establishment protocols is to develop distinct (unique, dynamic) keys each time the protocol is executed, anything less (static keys) makes the protocol insecure and open to attack. Key agreement can result from either key pre-distribution (i.e., key completely determined from seed values), or dynamic schemes. Dynamically generated keys, also known as session keys, support either key transport, or key agreement protocols. Session keys are normally generated from shared secrets and have a short useful life,

expiring after the communications session terminates. The cursory nature of session keys reduces the risk of cryptanalytic attack, less exposure of compromised cipher material, reduced requirement to cache key material (session keys generated only when needed), and each communications session exhibits different cryptographic features (which change the effectiveness of attack mechanisms). Key update protocols (e.g., key derivation) are related to key establishment services, in that they provide entities with a secure means of changing key material.

Recall that authentication protocols function to guarantee the identity of those entities one is communicating with. While key establishment (transport & agreement) protocols function to institute a joint secret among the communicating entities. Authenticated key establishment protocols combine the features of the above two protocols to create common secrets among identifiable network peers. These protocols have a collection of attributes that effect the selection and application of one protocol over another.

Meaningful protocol characteristics should be considered during any process that selects a key establishment scheme, several of the more conspicuous features are summarized in the following discussion. The

fundamental character of authentication methods confirm user key receipt, authenticate the key, and/or authenticate the user, these authentication features should be clearly recognized and mapped to the network security requirement(s).

Authentication methods may provide for one-way (unilateral), or mutual entity identification. The added security mutual authentication provides is generally paid at the price of increased message overhead (i.e., supplementary exchanges required to complete the authentication in the opposing direction[s]).

Key freshness and key control relate characteristics connoting key uniqueness with predictability of the key generation technique. Fresh (unique) keys preclude the possibility of an old key from being reused (by an attacker, or an unauthenticated user). Key control describes which entity in the key establishment protocol determines the resulting value of the key. Recall that in transport schemes, one entity determines the key and transmits it to the other entity(s). Agreement schemes provide a collaborative method in which two (or, more) parties can mutually generate a key that neither party can predict separately.

Protocol requirements (or non-requirements) for a trusted third party can reduce the entity authentication task (while creating a potential bottleneck at the trusted server) and shift workload to intervening nodes that provide on-line mediation of authentication exchanges. On-line trusted servers facilitate responsive counter measures to security attacks and key compromise, while off-line servers minimize the bottleneck potential with decreased capability to response to key compromise.

Related to trusted third parties is the type of key certificate (and initial certificate distribution) to be used. Symmetric key certificates are generally encrypted by the server and require real-time decryption (by the server), which may become a bottleneck, compared to public key certificates that can be publicly (unciphered) stored off-line on the server.

An attribute that spawns much interest is protocol efficiency. Efficiency is effected by factors such as the number of message exchanges (passes) to complete the protocol, overhead (number of message units to transmit) required, protocol algorithm execution time (a result of computational complexity), and the ability to pre-calculate and store parameters off-line, vice performing on-line interactive computations. Efficiency of cryptographic

algorithms and the resulting level of security can be directly related when rudimentary protocol calculations are repeated to obtain desired levels of security. As the calculation efficiency increases, so that more calculations can be performed in a given time interval, so increases the resulting protocol security valuation.

Secret key (symmetric) encryption techniques can be used with, or without trusted third party servers, or with timestamps (or nonces) to support secure key transport over open channels. Point-to-point (session) key update protocols and the Shamir no-key algorithm are examples of symmetric techniques that support secret key transport. One form of the point-to-point update method employs long-term keys that pre-exist at each entity. These long-term (key-encrypting, or key-transport) keys are near the top of the key hierarchy (just below master keys, and above short-term data/session keys). These keys can be distributed by out-of-band methods (e.g., courier, physical receipt, etc.), or in-band via active communications channels.

Typically point-to-point update protocols use an in-place symmetric (secret) key-encrypting key(K) to secure the transport of the new short-term session key(ki) from the initiator of the key update. In its simplest form the protocol can be accomplished in a single pass (one message

exchange from initiator[I] to receiver(R), e.g., I --[E(ki)K]--> R, where E is a symmetric encryption function, (ki) is the updated key from I, and K is the pre-existing key-encrypting (longer term) key used in the encryption function). This method is efficient, but not resistant to a replay attack on the initiator, and doesn't provide the receiver with key authentication, integrity, or freshness. Additional information fields (intended receiver ID, a redundant quantity, and a timestamp) can be added to the message to eliminate these short comings (e.g., I --[E(ki,ti,R)K]--> R, where ti is a timestamp from the initiator, and R is an ID of the receiver).

There may be times that both parties are required to provide inputs to jointly create a new session key. Collaborative key generation adds another exchange to the protocol and increases the protocol to two passes. Keyed one-way (hash) derivation functions are used with inputs from both parties, or a single party, to create the resulting session key.

If timestamps (and related clock synchronization requirements) are undesirable, nonces (random, sequential, etc. numbers) from each entity can be added to the messages, this will add another message to the exchange for a total of three messages to complete the protocol. Another method,

Shamir's no-key protocol, results in a secret key (K) shared by both the initiator(I) and the receiver(R).

Shamir's no-key protocol uses modular exponentiation with symmetric encryption to accomplish key establishment in three passes. A prime number is chosen one-time and securely distributed. The prime number (p) is selected so that it disallows calculation of logarithms mod p and is shared with both participating entities. Each party (initiator and receiver) selects a secret, random number (r[i] and r[r]) and stores it for later use. Once this setup is accomplished, the protocol may be performed. The initiator(I) starts the protocol by selecting a session key(K) and uses his secret random number(r[i]) to raise K to the modular power of r[i], mod p (K**r[i] mod p, where ** signifies exponentiation) and sends this message (number 1) to the receiver(R). The receiver takes the received value and raises it to the power of r[r] (R's secret number) and sends the result ((K**r[i])**r[r] mod p) as message number 2 to the initiator. The initiator(I) receives message number 2 and raises it to the inverse power of r[i] mod p-1, leaving the following result, K**r[r] mod p. The initiator sends this result, as message number 3 to the receiver(R). The receiver(R) raises the received value to the inverse power of r[r] mod p-1, leaving K mod p, which completes the

key exchange from initiator to receiver. The Shamir no-key transport protocol facilitates the same function as the Diffie-Hellman, key agreement protocol, i.e., secure, unauthenticated key establishment. The difference is that the Shamir exchange determines the session key(K) before hand by the initiator, and is completed in three passes. The Diffie-Hellman key agreement exchange constructs the session key by inputs from both parties, and only requires two passes.

The Diffie-Hellman key agreement protocol (shown in Figure 14) was the first efficient method for two parties to

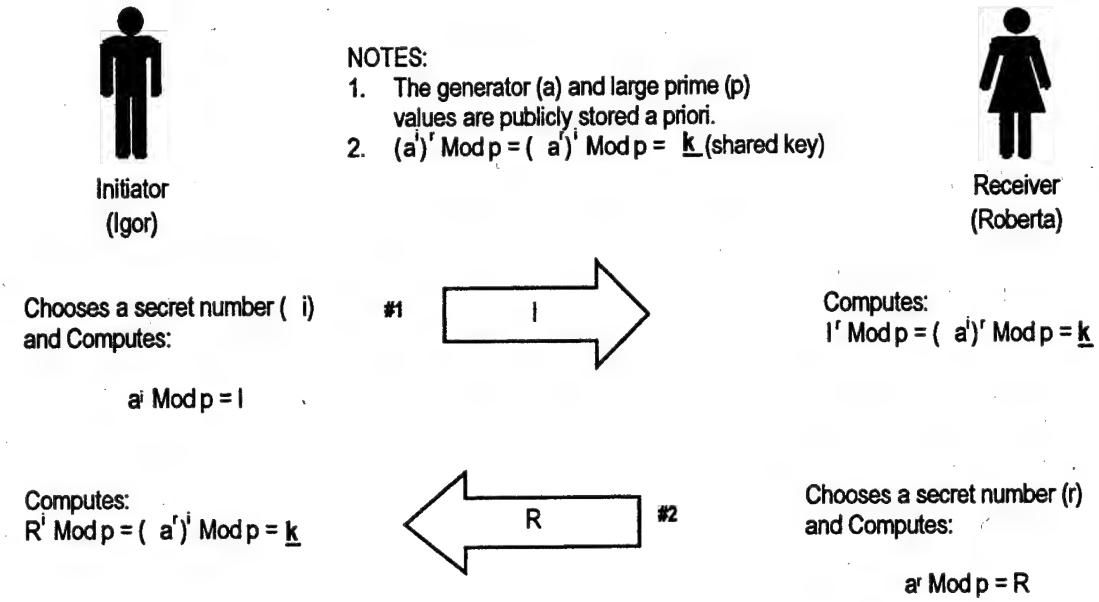


Figure 14. Diffie-Hellman Key Establishment

establish a secret (session) key with each other over an open channel. The basic security of the method resides in

the intractability of computing discrete logarithms. As with the Shamir scheme, a one-time prime number(p) and generator values (a) are chosen so that the generator(a) is equal, or greater than two, and less than, or equal to the prime minus two [2<a<(p-2)], and these values are publicly stored. Each member of the key agreement protocol selects a secret integer number (i, for the initiator and r, for the receiver) between one and p-2. The initiator starts the exchange by computing the value of the generator(a) raised to the initiator's secret number(i) mod p (a**i mod p, i.e., the Diffie-Hellamn exponential), and sends this message (number 1) to the receiver. The receiver performs a similar calculation with its secret random number and sends the result (a**r mod p), as message number 2, to the initiator. The initiator uses message number 2 to compute the session key(K) by calculating K = (a**r)**i mod p, and the receiver performs a similar calculation on message number 1 to compute the same key(K) value, K = (a**i)**r mod p = (a**r)**i mod p. Authentication can (and should) be added to the exchange by employing digital signatures on the exchange of the Diffie-Hellman exponentials between initiator and receiver.

Basic Diffie-Hellman type protocols can be developed into 0-pass protocols, if each Diffie-Hellman exponential is

treated as the public key of its respective user and publicly distributed via signed certificates. Certificate distribution creates a user challenge to store and retrieve all keys and certificates of the parties for which secure communications is desired. Hence the employment of a trusted third party server which simplifies user key storage and retrieval tasks.

Needham-Schroeder (shared key), Otay-Rees and Kerberos key establishment methods are examples of server based key transport protocols that use symmetric encryption. Each of these protocols employ a trusted third-party server, called a key distribution center (KDC), or a key translation center (KTC). The difference between a KDC and a KTC is that the KDC supplies and distributes the session key, and the KTC relies on one of the entities (communicating parties) to generate the session key for the KTC to distribute to the other networked entities. KDC centralize key generation and KTC distribute key generation. Kerberos provides a good illustration for the following server mediated key establishment discussion.

Kerberos (version IV and V) implements a KDC (called the Kerberos authentication server [KAS]) to provide a session key and to authenticate the initiator (called the client) to the receiver (called the server). Kerberos also

provides data confidentiality and data integrity, the remaining discussion focuses upon the server mediated authentication and key establishment protocol features that Kerberos typifies.

Kerberos provides a session key and authentication in four exchanges, the first two exchanges are between the initiator (client) and the trusted server (KAS), and the last two exchanges are between the client and the receiver

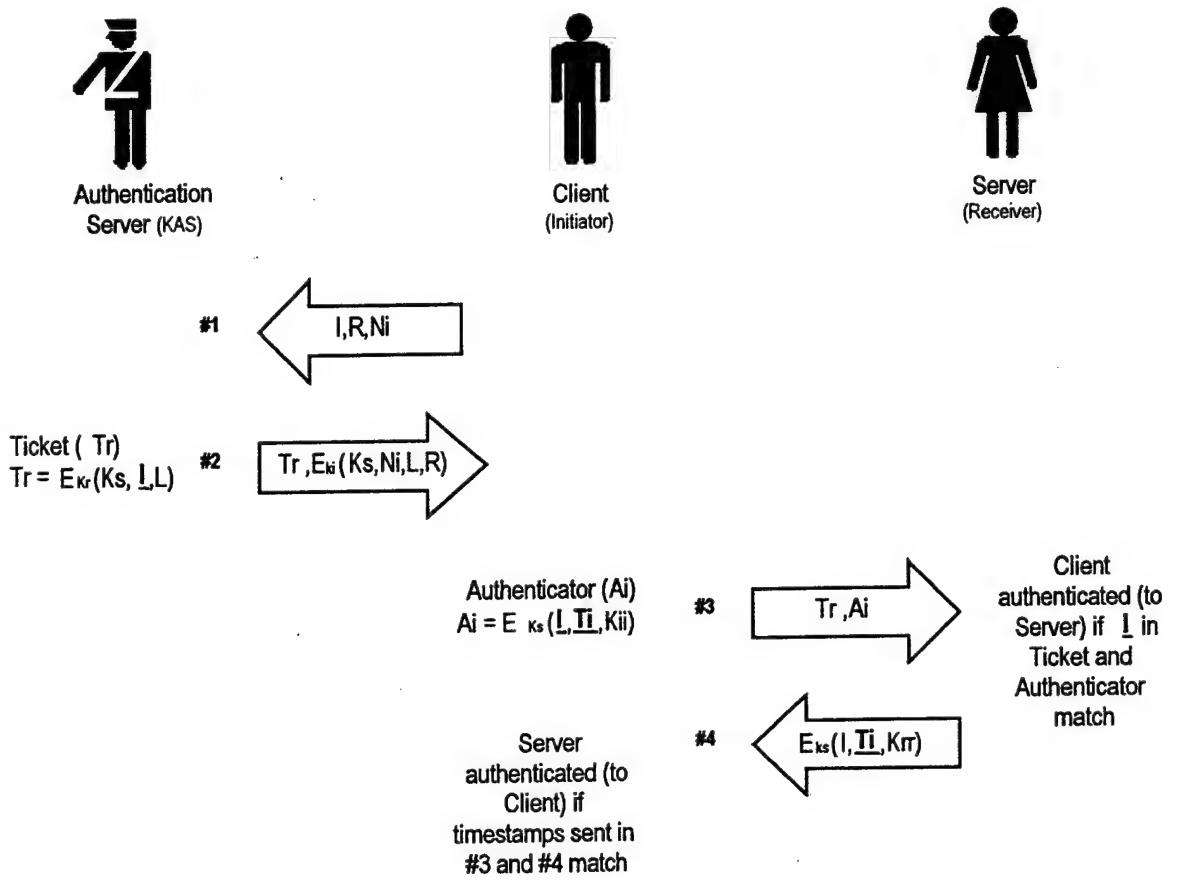


Figure 15. Basic Kerberos Authentication

(server). The client starts an exchange to the KAS by sending a message (number 1) with the client's ID(I), the server's ID(R), and a nonce (a non-repeated random value) generated by the client(Ni) [I --[I,R,Ni]--> R]. The KAS replies to the client(I) with message number 2, consisting of a ticket(Tr) for the server(R) containing the session key(Ks), the client's ID(I), and the valid lifetime of the ticket(L) encrypted by a key(Kr) shared between the server (R) and the KAS [Tr = E(Ks,I,L)Kr]; and Ks, Ni, L, and R encrypted by a key(Ki) shared between the client (I) and the KAS [E(Ks,Ni,L,R)Ki]. The client(I) passes the ticket(Tr) and an authenticator(Ai) on to the server(R) as message number 3. The authenticator(Ai), generated by the client(I), contains the client's ID(I), a timestamp from the client's clock(Ti), and an optional sub-key(Kii), and is encrypted with the session key(Ks) generated by the KAS [Ai = E(I,Ti,Kii)Ks]. The client(I) is authenticated one-way (unilaterally) to the server(R) when the server verifies the authenticator(Ai) in message number 3. The last message [number 4, from the server(R) to the client(I)] is optional and authenticates the server to the client (provides mutual authentication). This optional message contains the client's(I) timestamp(Ti) and an optional sub-key(Krr)

generated by the server(R), and encrypted with the session key(Ks) generated by the KAS.

The Kerberos protocol typifies the rigor and features that symmetric key, trusted on-line server mediated techniques employ to provide key establishment and authentication services. The main drawback to these hub type trusted third parties (KDCs and KTCs) is, that as the number of distributed users(n) increases, so to does the number of keys(Kn) (one shared between each user(n) and the KAS) increase as a factor of n squared (n**2). Asymmetric (public) key methods offer alternate solutions that mitigate the key numbers problem and permit user scalability without overburdening the trusted server with numerous key storage and retrieval operations.

Asymmetric (public) key establishment methods advantageously employ off-line trusted third party servers in a variety of functions ranging from Certification Authorities (CAs), name servers, registration authorities, key generators, and certificate directories. Only the CA directly communicates with the user. Name servers, registration authority, key generation and certificate directory servers provide functions that can be performed by the CA, or off-loaded to separate, distributed trusted servers (easing the load upon the CA).

The advantage of off-line operations is that they are completed before communication exchanges are initiated. This saves time waiting for operations to be processed in the serial fashion of user-to-user communications. Best use of symmetric and asymmetric techniques can be made when both techniques are used in hybrid schemes, using symmetric techniques for bulk data encryption sessions, and asymmetric schemes for key signature and management functions (Menezes, et al, 1996).

B. SECRET SHARING

Secret sharing methods are related to key establishment techniques and facilitate secret distribution among many users. These techniques provide distributed trust and shared control of crucial actions by a subset (n users) of the total(t) system users.

The basic concept is to divide a secret (key, control lock, etc.) into n distributed shares. Specific subsets (of n users) can combine their shares to reconstruct the secret. This concept could be implemented with the use of a trusted server that shares a secret with each user(n), and uses each of the n user inputs (shares) to construct the secret. This scheme gets more cumbersome as the number of users(n) increases, and the secret is not a truly collaborative (shared) effort of all the users, but rather mediated by the

trusted server, and the resulting key must be communicated securely to each of the (n) users.

Representative secret sharing implementations allow shared control and distributed, cooperative secret agreement for multiple sets of user groups. Simple two-party shared secret schemes use modular addition to generate the secret value. The protocol uses a trusted third party to select a secret number(S, between 0 and m-1), m (an integer known only to the trusted third party), and a value, S1 (between 1 and m-1). The trusted third party calculates the shares, (S-S1)mod m and (S1)mod m, and distributes the share values to the users (I and R). The two users enter their individual values (S1 and S-S1, respectively) into a tamper-resistant trusted device that combines the two values to obtain the secret(S), which grants control of a guarded operation, or secret.

Modular addition secret sharing techniques can be extended to n participants. The participants must combine all their individual share values (Si, where 1<i<[n-1] and one value St, where St=S-{summation[Si]}mod m) to obtain the secret key/control value(S). The approach described above splits control n ways, and requires all n shares to construct the secret key/control value(S). This is a special case (n,n) of the more general (t,n) threshold

scheme, where t is the number of shares required to construct the control/secret key(S).

A threshold scheme is considered 'perfect' when any party knowing t-1, or less, shares gains no knowledge of the control/secret key(S). One implementation of this type scheme is called Shamir's threshold scheme which uses polynomial interpolation together with the knowledge that a single variable (univariate) polynomial function [y=f(x)] of degree t-1 is uniquely defined by t points(x,y). Such a function produces t independent equations with t unknowns.

To implement this method the trusted server chooses a large prime(p); defines S (the key) to be the constant 'a(0)', (which is the value of f[0]); defines t-1 coefficients a(1) through a(t-1) of exponents x**1 through x**(t-1), respectively; calculates the shares S(1)[f(1)mod p] through S(t)[f(t)mod p]; and securely transfers the shares [S(1) through S(t)] to each corresponding user with share index (1 through t). When any 't' users pool (via a tamper-proof device) their shares(Si) and index(i) values, t equations with t unknowns [i.e., the constants a(0) through a(t-1)] result. The constants can be determined and the key(S) is obtained (by noting that S = a(0) is the session key). The characteristics of this method are:

* 'perfect' (t,n) threshold scheme (i.e., monotone access structure)
* 'ideal', share[S(i)] length equal to key(S) length
* Expandable to new users without impacting existing shares
* Multi-level user control when multiple shares given to users
* Security independent of unproven assumptions

Threshold methods can be expanded into generalized secret sharing schemes, that allow only authorized user share subsets to recover the key(S) when user shares(Si) are pooled (Menezes, et al, 1996).

Generalized secret sharing schemes facilitate pooling of authorized user share subsets(A), vice unauthorized user share subsets(U), to recover the key(S). The authorized user share subsets(A) comprise an access structure of the total set of users(T). Generalized secret sharing schemes exist with extended features, some of the more interesting are:

* Pre-positioned shared secrets (all required shares, but one, are distributed to user nodes, so that a single share(Si) can establish the key(S) at the participating user nodes)
* Dynamic secret sharing (a pre-positioned method that allows the key(S) to be changed according to the value of the establishing share[Si])
* Multi-secret (methods in which different keys(S) correspond to different user share subsets)

Secret sharing methods may appear similar to conference keying schemes, but secret sharing methods display some noteworthy differences:

* They must securely transfer share information to participating users
* Shares are pooled (combined) in a tamper-proof device to obtain the key(S)
* Same key(S) results every time shares(S[Si]) are pooled (static session-to-session)
* Keys(s) associated with share sets (vice sets of user identities)

Conference keying generalizes two-party key establishment methods among three, or more participating users. These techniques are conducted over open channels, employ dynamic session keys, obtained individually by each participating user, and a user group can be associated with a particular session key(S). An apparent solution is to employ a trusted server holding a different symmetric key with each participating conferee. For each session the trusted server would generate the session key(S) and distribute it to all conferees via each conferee's individual symmetric key.

This solution presents a few obstacles to efficient network operation. The server is required to operate on-

line, in real-time with each communication exchange, and the server will become more computationally burdened as the number of conferees is increased.

This scheme could be modified so that a 'designated conferee' could assume the key generation/distribution functions and eliminate the need for a trusted server. The designated conferee could distribute session keys to each conferee via Diffie-Hellman exponential pairs held between the designated conferee and each participating conferee. This method is still not easily scalable, because the computational burden is just shifted from the trusted server to the designated conferee.

A better alternative is to distribute the burden among all the (n) participating conferees. A method of achieving conference keying has been proposed in the form of the Burmester-Desmedt protocol. The Burmester-Desmedt protocol sequentially orders (0 to n-1) all (n) conferees; requires each participating conferee to compute its individual Diffie-Hellman exponent (a**r[i]) mod[p]), using an individually selected random value(r[i]); and compute a quotient. The quotient consists of the conferee predecessor's Diffie-Hellman exponent(a**r[i-1]) divided by the conferee successor's Diffie-Hellman exponent(a**r[i+1]), raised to the exponent of the conferee's random value(r[i]).

All (n) participating conferees send their Diffie-Hellman exponent(a**r[i]) and the value of exponentiated quotient(EQ[i]) to the other (ordered 0 to n-1) participating conferees. The group session key(S) is then computed from these received values individually by each conferee (see Menezes, 1966 for details).

Modified versions of the above Burmester-Desmedt broadcast conference keying methods allow for each conferee to only exchange keying information with its immediate predecessor(i-1) and successor(i+1) conferee.

C. KEY EXCHANGE ATTACKS

Authenticated key establishment protocols, whether party-to-party, or conference (multi-party) require care in their design so that they are not susceptible to common forms of active attack. The more frequently discussed forms of active protocol attack found in the literature are (Menezes, et al, 1996):

* Attacker-in-the-middle (eavesdropper)
* Replay
* Interleaving
* Mistrusted server

An attacker(A) (interloper, eavesdropper, intruder, etc.) may intercept an unauthenticated Diffie-Hellman key agreement exchange from an initiator(I) intended for a receiver(R), replace the initiator's(I) Diffie-Hellman exponent (a**r(i) with its own Diffie-Hellman exponent [a**r(a)], and send it on to the intended receiver(R). The

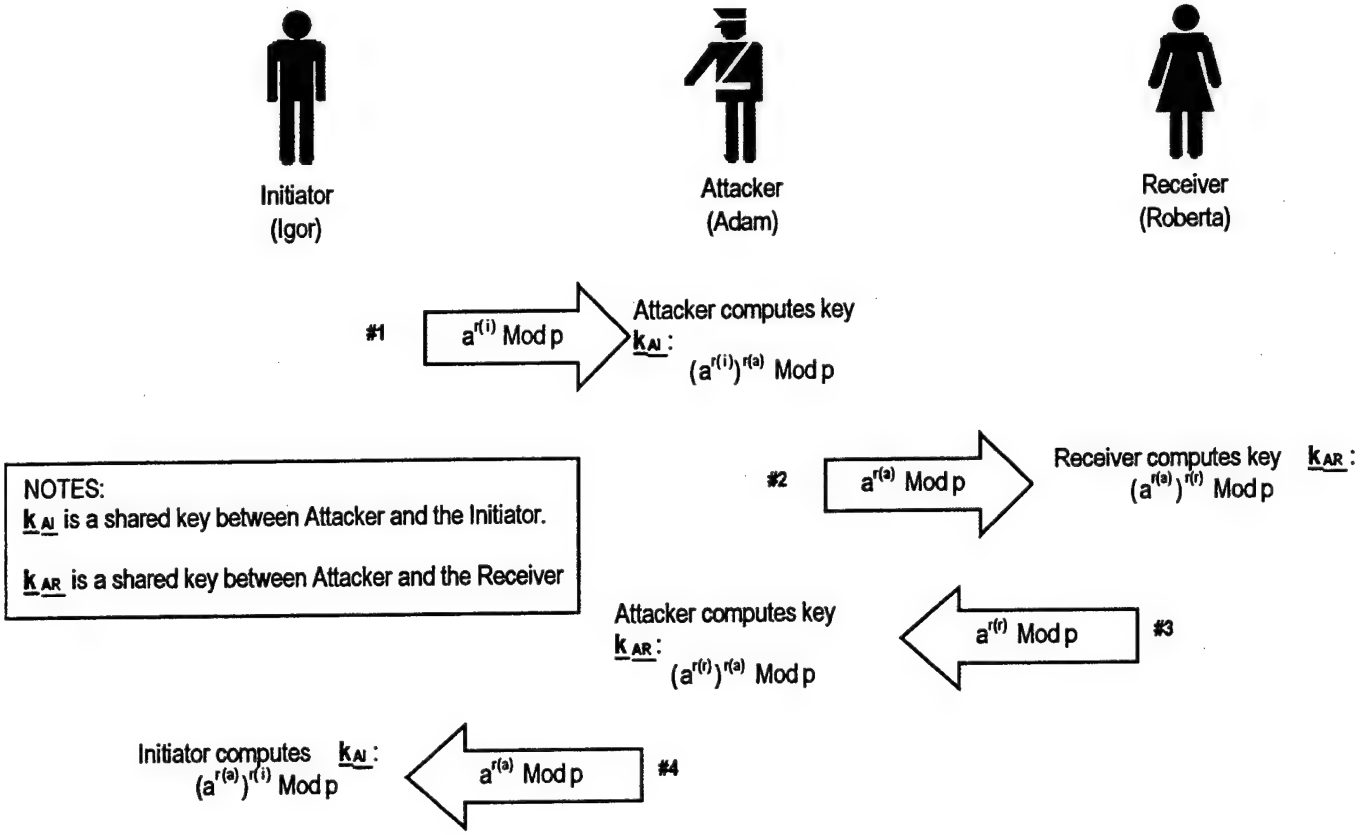


Figure 16. Man-in-the-Middle Attack

receiver will reply back to the attacker with its Diffie-Hellman exponent (a**r(r), and the attacker will again

replace the receiver's Diffie-Hellman exponent with its own and complete the exchange with the initiator. The result is that the attacker will be between the initiator and the receiver passing and observing information between the two parties undetected (as shown in Figure 16). The protocol flaw to avoid is always authenticate the identity of the party one is communicating with.

Successful replay (or reflection) attacks, discussed earlier, result in the attacker impersonating a receiver(R). Knowledge of a secret (e.g., key) common to an initiator and a receiver used to authenticate the parties to each other provide the following review of a reflection scheme.

The initiator starts the protocol by sending in a random value(r[i]) to the receiver(R), which the attacker(A) intercepts and replays back to the initiator as the start of an independent second authentication exchange. The initiator replies to the attacker's exchange (the second authentication exchange) with another random value(r[i*], different from the previous one) and encrypts both random values with the secret key shared by initiator(I) and receiver(R) (E{r[i], r[i*]}). The attacker again replays the encrypted exchange (E{r[i], r[i*]}) back to the initiator as the reply to the first exchange from the initiator. The initiator receives the replayed encrypted

random values, recovers the initial value [(r[a]), from the first exchange] and the accompanying value [(r[i*]), of the second exchange], and completes the first protocol exchange by sending value r[i*] to the attacker. The initiator(I) presumes that the value sent, by the attacker, was the receiver's random value (i.e., r[a] = r[i*]) and that it is communicating with the receiver(R), but in reality the attacker has successfully impersonated the receiver to the initiator, and the second protocol is never completed.

The reflection ploy can be defeated if different symmetric keys are used for encryption by each party. Use of asymmetric keys would only require that additional public keys, vice secret keys be known. An identifier of the encrypting party can be placed within the encrypted portion of the message to break message cryptographic symmetry (Menezes, et al, 1996).

In the two protocol exchanges above, one of the authorized parties initiated the exchange. The next attack discussed is similar to the reflection attack, except that the attacker initiates a protocol exchange with both parties (the initiator and the receiver).

The interleaving attack is started by the attacker (pretending to be the initiator) to the receiver in the first protocol exchange, and the attacker pretending to be

the receiver to the initiator in a second protocol exchange. The attacker sends a random value(r[i]) to the receiver and the receiver replies directly back to the attacker (mistaking attacker as legitimate initiator) with a correct digital signature and random value (r[r]). The attacker next initiates a second protocol (masquerading as receiver) with the initiator, passing the random value(r[r]) (obtained from the receiver) to the initiator. The initiator replies to the attacker with a correct digital signature and random value (r[i*]), which the attacker passes to the receiver, to complete the first protocol. The random numbers (r[i], r[r], r[i*]) used in the protocol were for the purpose of ensuring that replies (nonces) are fresh (unique). However, additional protocol information (i.e., correct sequence, time information, or binding a party identifier to the signature) is needed to ensure that one protocol exchange sequence can not be used to complete another protocol exchange sequence (Menezes, et al, 1996).

Attackers can be legitimate system users, strike from within trusted server systems, and impersonate other system users to the trusted server. A resident system attacker can intercept a protocol exchange from a receiving party to the trusted server and substitute its identifier(A), nonce(Na) and server key(Ka) for the initiator's identifier(I),

nonce(Ni) and server key(Ki). The trusted server will reply to the attacker with the session key(S) and nonce(Ni) encrypted with the initiator's private server key(Ki), to be forwarded to the initiator(I) by the attacker, and the session key(S) and attacker's nonce(Na) encrypted with the attacker's server key(Ka). The attacker can successfully impersonate a receiver(R) to the trusted server (and ultimately, to the initiator). This is possible when the initiator relies upon its nonce(Ni) to indirectly identify (via the trusted server) the receiving party, and the trusted server fails to correctly check the clear text identity and nonce fields with the encrypted fields of the attacker's (impersonated) exchange (Menezes, et al, 1996).

The four attacks discussed above are fundamental to all exchanges, and require careful consideration when selecting exchange parameters (e.g., identifiers, nonces, timestamps, random numbers, etc.) to support the desired security level that protocol execution is expected to guaranty.

V. NETWORK (MULTICAST) SECUTITY IMPLEMENTATION

This chapter will discuss security considerations appropriate to multicast protocol implementations for tactical data networks (TDN) and networked Naval applications. TDNs will be defined in a Naval setting, and user requirements for these data networks/applications will be reviewed. The discussion of tactical networks/Naval applications will frame the treatment of security services to incorporate into multicast protocols in open networks.

A. TACTICAL DATA NETWORKS (TDN)

TDNs are the principal communication networks of deployed Marine Air Ground Task Forces (MAGTF). MAGTFs are sized in terms of a division (i.e., Marine Expeditionary Force[MEF]), regiment (i.e., Marine Expeditionary Brigade[MEB]), or battalion (i.e., Marine Expeditionary Unit[MEU], and composed of ground, air, support, and command (i.e., Headquarters[HQ]) elements. The MAGTF command element is connected to other strategic nets (Defense Information Systems Network [DISN]) and joint task [JTF] force component HQs (i.e., Army, USAF, and Navy) by satellite (wireless/limited bandwidth) connectivity. The MAGTF command (HQ) element is connected the ground, air and support elements by satellite, multi, or single channel

radio links. These wireless links are restricted to 16, or 32 Kbps, depending upon equipment providing link access (Petitt, 1996). The tactical data network consists of Ethernet local area networks (LANs) supported by servers, and connected to external networks (i.e., DISN strategic nets and JTF component HQs) through Gateways. The LANs support several tactical data systems, such as:

* Tactical Combat Operations (TCO)
* Intelligence Analysis System (IAS)
* Advanced Field Artillery Tactical Data System (AFATDS)
* Marine Combat Service Support Command and Control System (MCSSC2)

Field commanders and their operations officers to obtain a near real-time comprehensive view of the battle-space primarily use TCO. Video teleconferencing is planned for incorporation, and distributed collaborative planning (e.g., white boarding) may also be added to the TCO system (Petitt, 1996).

IAS provides automated support for the collection, processing, production and dissemination of intelligence data within the MAGTF. Intelligence data can be distributed (pushed) to recipients without their request, or downloaded (pulled) from centralized servers.

AFATDS in an automated command and control (C2) system that coordinates artillery units using time critical fire support data. And MCSSC2 is a logistics support system for the MAGTF support elements. These systems, described above, typify the most essential requirements of tactical networks and their function within deployed forces.

B. NAVAL DATA NETWORKS

Naval forces also utilize data networks to move, process and exchange information. Typical functions supported by Naval networks are (Rao, et al, 1997):

* Text Messaging/Email
* File (data/image) Transfer
* Broadcast (voice/data/video)
* Video Teleconferencing (VTC voice/video)
* Interactive (collaborative) Planning
* Real-time (hard and soft) Data Transfer

The items enumerated above describe Naval network requirements from a functional usage viewpoint.

Naval messaging and Email are common forms of information transfer that can have a range of delivery priorities and message security classifications. Both terrestrial and wireless transmission media support messaging and Email traffic. Defense Message System (DMS),

Joint Maritime Command Information System (JMCIS), and Global Command and Control System (GCCS) applications are examples of military applications that support Text Messaging/Email functions.

Data/image file transfers consist of non-real-time bulk data transfer that typically fit into client-server architectures. The receiver is the client of the transmitter e.g., downloading of weather maps from an archive site.

Broadcast of voice and video are non-interactive in nature, and require no reply exchanges from receivers. This functional category is less demanding on supporting network architecture than interactive video teleconferencing (VTC).

Video teleconferencing increases the network burden of simple broadcasts by making all parties transmitters with increased emphasis to minimize transmission latencies.

Interactive (collaborative) Planning is a dynamic multi-media form of VTC placing maximum demands upon the network infrastructure.

Real-time (soft and hard) data transfer can either refer to time sensitive (soft) applications, or time critical (hard) applications.

A soft real-time (also referred to as near real-time) application is one that has specified response times between

program inputs and outputs. The response times are values within an acceptable range about an average value. Therefore, a response can be late without considering the application to be in error. An example of a soft real-time application is the Automated Teller Machine (ATM). The application must respond to consumers within an acceptable time otherwise the ATM application will be considered unreliable and lead to customer dissatisfaction. A single transaction is not a major problem if the response is 10%, 25% or even 50% slower than the specified average value. The ATM is an example of a slow, soft real-time system. A video conferencing is an example of a fast soft real-time application. Video conferencing applications must display video with high frame update rates. An odd "glitch" that might appear can be accommodated as long as the system maintains a reliable connection.

A hard real-time application has a specified response time (maximum or minimum) that is an absolute value. When the specified response time is not met the application is considered to be in error. When in error, the application is required to perform some type of recovery, continue with reduced functionality, or shutdown. In extreme cases this can lead to damage, loss of equipment, injury, or loss of life. Examples of hard time-critical applications are

nuclear power plant controls (aboard ships, or submarines), aircraft flight controls, sensor systems, and combat data applications. Hard real-time applications have absolute deadlines in which a response must be elicited otherwise serious consequences may result.

C. REQUIREMENT SUMMARY

The requirements discussed above can be satisfied with two basic types of transfer modes, batch and streaming. Batch mode transfer facilitates text messaging/email and file (data/image) exchanges. Broadcast (voice/data/video), video teleconferencing (VTC voice/video), interactive (collaborative) planning, and real-time (hard and soft) data exchanges can make use of streaming transfer technologies. The number of network participants will effect tolerable ranges of delay, error rates, and suitability of the multicast technique to best employ. Interactive multi-media collaboration is likely to be the most demanding information transfer application for a multicast network to support.

Security services are required for all military information transfers as well as financial and commercial transactions. Civilian use of security services is rivaling governmental use and sophistication of security implementations. Commercial off-the-shelf (COTS) implementations are becoming more practicable solutions for

governmental (i.e., military) security requirements. The main difference and central issue of civil and governmental security systems is the allowed key length of civilian versus governmental crypto-systems.

The remaining discussion assumes that civil (e.g., financial and commercial) crypto-systems are equivalent to governmental (e.g., military) crypto-systems with the exception of key length related computational and block size implications. When key lengths are equal, implementation considerations become the same for similar types of information transfers.

When selecting appropriate network data security services, basic consideration must be given to the type/usage of data, and the heterogeneous characteristics (e.g., quality of service, bandwidth, S/N, etc.) of the network(s) that the data will be distributed over. A "one size fits all" security implementation is as mythical as a "do it all" multicast protocol. Security implementation (and protocol selection) is an optimization of features that compliment data type and usage.

The requirements discussed group into four basic types by usage:

* Text/messaging (store & forward)/file transfer
* Broadcast (Non-inter-active) voice/video

* Multimedia (VTC)

* Real-time (e.g., collaborative [inter-active] planing, tactical command & control[C2]) information transfers,

And two basic types of transfer mechanisms used are:

* Bursty (e.g., batch text/messaging/file) transfers

* Streaming (Real-time, Multimedia, voice/video) transfer,

Discussion of end-to-end security services, for the above stated data exchange requirements, will focus on transport layer implementations. Transport layer security implementations provide standard, consistent network security provisions that are transparent to applications running at individual network nodes. Commonality of security primitives (algorithms) is achieved for all multicast receivers and this helps to maximize network performance.

D. SECURITY SOLUTIONS

Authentication is the first security parameter that must be fulfilled for any data transfer. Elemental to any exchange of data is the identification and authentication of the communicating parties. Bursty information transferred via broadcast, text/message, or files (i.e., generally one direction exchanges) requires that, minimally, the sender

authenticate to the receiver. This authentication ensures that the receiver is not being spoofed by a masquerade attack. The receiver also needs assurance that data, received from the sender, has not been altered in transmission, storage, or retrieval by an intruder (i.e., data integrity is maintained). Data integrity and one-way authentication can be satisfied with digital signature schemes with appendix (Menezes, et al, 1996). This type of digital signature scheme requires the original message as input into the verification operation.

One-way hash functions form the basis of signature schemes with appendix, and can be applied to messages of arbitrary length. After the hash of the message is computed, the hash value is encrypted with a public key algorithm (e.g. RSA, etc.) and sent, with the original message, to the receiver(s) for verification (Schneier, 1996). Use of public key encryption (of a message digest) relieves the key storage and distribution burden implicit in symmetric key schemes. Late joins to the exchange would not introduce additional requirements to receive a secret key in order to process transferred information.

Interactive (two-way), real-time multimedia exchanges require the transport protocol incorporate mutual authentication mechanisms. Receiver controlled multicast

protocols have shown better performance avoiding the packet acknowledgment (ack) implosion problem. These types of protocols can most efficiently implement mutual authentication services, especially, those multicast protocols that distribute ack processing to receiving nodes acting for the sender, within smaller localized receiver sub-regions. These nodes (called designated receivers in RMTP) function as both sender and receiver and are structured in a hierarchical order. Employing these designated receivers to assist in authentication of receivers to the sender, and vice-versa has the potential to reduce the burden on the sender. Authentication tree structures of designated receivers is one method to distribute the sender's authentication burden.

Authentication trees (first proposed by Merkle) are binary trees that allow public values (e.g., keys) to be verified through successive hashing operations as the tree is traversed from a leaf node (receiver) to its root (sender). Only one value (the root value, vice all leaf node values) requires registration with a trusted certification authority (CA).

Once parties (i.e., the leaf nodes) are authenticated to their respective designated receivers, the process can be repeated at successively higher levels in the multicast

distribution tree, until authentication is ultimately made with the sender (at the root of the distribution net). The advantage is that the designated receivers act independently, and in parallel to shorten the overall time to authenticate the multicast tree. Authorized multicast membership is ensured because the hash value calculations from leaf node to the root are always compared to the root node value. When there is no match, authentication does not exist for that designated/receiver. Late multicast member joins may be accommodated by constructing the authorized multicast distribution tree a priori before any network activity is started (e.g., pre-planned network, like a tactical data net, developed from a communication plan).

Other authentication mechanisms must be sought to provide efficient security services for networks requiring dynamic multicast membership capabilities. Trusted server methods (e.g., X.509, Needham-Schroeder, and Kerberos) offer the flexibility which multicast parties require. Additional cache memory would allow the trusted server to recognize membership in different multicast groups while providing a minimum number of session keys (vice n**2 keys per multicast session) to accomplish these diverse multicast sessions.

Public key encryption methods offer the benefits of entity authentication and confidentiality in fewer

exchanges. Authentication/key establishment schemes (e.g., X.509) can operate with specified encryption schemes, but generally default to the RSA public key standard. RSA is known to be several orders of magnitude more computationally expensive than more conventional symmetric (e.g., DES) encryption methods.

Elliptic curve cryptosystems (ECC) are another class of public key algorithms that rely upon the difficulty of solving the discrete logarithm problem (given g & y, find x in: y = g**x [mod p]) over the points on an elliptic curve (Certicom, 1997). This method (ECC, an ANSI X9.62 draft) has been reported to be bit-for-bit stronger than RSA methods, and for the same security has an order of magnitude less computational expense.

X.509 is part of the X.500 series of recommendations that define a directory service. These recommendations are expected to become widely employed and can specify ECC encryption methods (vice RSA schemes) to operate with reduced overhead penalties to the multicast session. (Stallings, 1995). The public-key user certificate is the central feature of the X.509 authentication scheme. The certificate contains: certificate format version, certificate serial number, signature algorithm identifier, issuing certification authority (CA), valid period,

certificate user identity, user's public key and public key algorithm ID, and the signature of the above certificate fields encrypted with the CA's private key. For multicast application of X.509, designated receivers should assume CA/directory functions for those receivers underneath each respective designated receiver. CA chains (similar to a leaf-root path of an authentication tree) can be used to provide authentication of multicast members in disparate sections of the multicast network. X.509 can provide scalable authentication services (i.e., one-way, and mutual authentication) with and without the requirement for use of synchronized clocks by sender(s) and receiver(s).

Authenticated, untampered data streams may satisfy commercial and financial data exchange requirements, but all Naval multicast users additionally require confidentiality of transferred information. This essential security service is most efficiently provided to common forms of data exchange via secret key cryptographic algorithms, such as DES, or IDEA. Reports have been made that 128Kbps video streams have been DES encrypted and processor load was 1% of the capacity of a Sun Sparc 20 workstation (Van Jacobsen, 1994). Voice and video streaming applications require that latencies, whether network or security service induced be

minimized, and present encryption technology efficiencies favor the use of symmetric algorithms for confidentiality.

Streaming (one-way) types of broadcast information exchanges would also benefit from the use of secret key encryption, in place of the public key schemes to ensure confidentiality because of the smaller, more numerous data cells used in streaming protocols, and the orders of magnitude increase in execution speed secret encryption algorithms enjoy. DES encryption rates (in C implementations) of 3.5 to 14.5 Mbps have been reported in the literature (Van Jacobsen, 1994).

An interesting alternative to encryption has suggested packet scrambling at the transport layer to achieve data confidentiality, without the overhead of conventional encryption algorithms (Shiroshita, et al, 1996). A scrambling key generates new packet sequence numbers for the packet stream. The de-scrambling key is generated from the scrambling key by the receivers and used to determine the correct order to reassemble the data packets. Small data packet protocols (e.g., streaming, ATM, etc.) would receive the most benefit from packet scrambling because of the longer packet sequences.

Unique schemes such as packet scrambling may offer attractive solutions, but one should not overlook proposed

Internet (open system) standards (i.e., IPv6). Since 1994 a large body of work has been, and is still being conducted on a package of related standards (i.e., RFC's 1752, 1883, 1886, 1971, 1993, etc.) that collectively define a new (or next generation) Internet IP protocol.

The proposed Internet protocol provides for authentication and data integrity through the use of an authentication header (AH) and equally provides for data confidentiality with the use of an encapsulated security payload (ESP). Algorithms for the AH are not specified, but all IPv6 implementations are required to support MD5 (RFC 1828) when no other authentication algorithm is identified. Similarly encryption algorithms are not specified for use with the ESP, but implementations of the protocol must support the data encryption standard (DES, RFC 1829) cipher block chain (CBC) mode when no other security associations have been made.

Key management provisions (for the MD5 AH key, and ESP key) are provided separately from the protocol itself. This independence allows key management protocols to be implemented without effecting the basic operation of the security features of the network protocol (IPv6). RFC 1825 defines the key management features that all IPv6 implementations must support. Manual key management and an

Internet standard key management protocol are to be accommodated under the definitions of RFC 1825.

Manual key management allows the key management system's keys and the communicating host's keys to be manually configured. This method does not scale well. At present there is no Internet standard key management protocol, though work on this topic is under way.

Whatever key management protocol is developed, it is generally accepted that an Internet-wide public-key infrastructure will be required. IPv6 has been designed with necessary security options and multicast features for open systems environments. The allure of universal interoperability may subdue the quest for an optimal secure multicast protocol, and an Internet standard, of itself, may be the best answer.

VI. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

Traditional Naval information transfers (i.e., messaging, voice, and data files) have enjoyed encryption at the cost of specialized equipment and interfaces inserted into restricted (private) communications channels. A new paradigm of secure communications in open (shared) networks using contractor-off-the-shelf (COTS) equipment is at the forefront revolutionizing the manner in which future interactive multi-media transfers will be serviced. Multicast offers an efficient dynamic sharing of bandwidth. The judicious addition of security services to multicast techniques will yield a secure multicast that is capable of supporting the requirements of the new information technology paradigm for the 21st century (IT 21).

1. Authentication/Integrity/Confidentiality

Any security service scheme employed in open network environments must bundle the essential three security primitives (authentication, integrity and confidentiality). This basic security package should provide the user with an integrated and flexible set of security options that accommodates varied requirement sets from clean optical

quality networks to noisy wireless tactical data network environments.

2. Security Service Implementation

The debate will not end here about what type of security design philosophy is best (built-in, or added-on). However, the advantages of embedded security services must be stated and not glossed over. A major benefit is the multi level security (MLS) possibilities that integral security primitives make plausible. MLS capability alone peaks much interest and consumes a large amount of effort within security organizations. Security options incorporated within the network protocols relieve the computational burdens on higher level applications, and provide a means to enforce a consistent network-wide security policy. Transport/network layer protocols are generally accepted as the most prominent locations to implement protocols that facilitate the three basic security primitives (i.e., authentication, integrity and confidentiality). Large/multicast networks offer the potential to distribute the security overhead among many nodes vice overloading a single source, and reduce the possibility of a single point failure.

3. Key Management

Efficient key establishment/distribution/management schemes and infrastructure are the central challenges to building scalable interactive secure multicast networks. Secure Session keys are most probably supported by public-key infrastructure (PKI), i.e., trusted servers that provide requisite in-band key distribution and management services to authenticate network participants. Automated in-band key management techniques should also be symbiotic with manual key management procedures.

B. RECOMMENDATIONS

1. Incorporate open system standard security protocols

IPv6, streaming data protocols and ATM backbones are the generally accepted methods of transporting information to/from workstations in the next room, or neighboring continent. Continuing efforts must be expended to incorporate and evolve standards in concert with technological advances. Only widespread adoption of these open standards makes interoperable COTS supported architectures and infrastructures possible. Virtual private networks (VPNs) are an example of new technology, which should be evaluated and incorporated into the open standards base. VPNs offer organizational groups the ability to use public networks (i.e., Internet) to securely transfer

information that would otherwise be transmitted via expensive private and/or leased secure media.

2. Investigate scalable In-Band Key Management Schemes

Key establishment/agreement protocols generally employ trusted third party (TTP) infrastructures. Reducing the need to provide different key management methods for each user/application within, or among networks will distribute and simplify the key management infrastructure burden. Such key management consolidations will enhance scalability of network key management services. The Internet security association and key management protocol (ISAKMP) draft (March 10, 1998) and RFC 1949 (Scalable Multicast Key Distribution) are illustrative endeavors to investigate open key management standards. These and other key management investigations should be vigorously supported.

3. Perform Algorithm Performance vs. security strength assessments in functional network environments

Algorithms (e.g., SHA, MD5, DES, IDEA, RSA, El Gamal, Elliptic Curve Crypto-systems[ECC], etc.) are the cryptographic elements of all authentication, confidentiality, integrity or key management protocols. Assessments of these algorithms (with respect to computational expense and the resulting security stamina each algorithm provides) should be accomplished within functional network environments. With these comparisons, as

a relative performance baseline, other non-cryptographic schemes (e.g., packet scrambling) may be assessed for effectiveness as alternatives to traditional cryptographic methods. The search for more computationally efficient security algorithms is an ever-present activity and requires a security cost/benefit baseline be established so that meaningful future network security performance comparisons can be made.

APPENDIX A. SECURITY-RELATED WEB SITES

Security-related Web sites, discovered while doing research for this thesis, are listed below. The title and Uniform Resource Locator (URL) for each Web site are shown.

6bone
http://www.6bone.net

A Security Architecture for Fault-Tolerant Systems
http://cs-tr.cs.cornell.edu/Dienst/Repository/2.0/Body/ncstrl.cornell%2fTR93-1354/ocr

Algorithmic Research (AR) a global supplier of cryptographic data security solutions
http://www.arx.com/html/about.html

An Architectural Overview of UNIX Network Security
http://www.alw.nih.gov/Security/Docs/network-security.html

Aventail: Virtual Private Networks (VPN)
http://www.aventail.com/

Bibliography on Authentication Codes
http://bibd.unl.edu/~stinson/acbib.html

Caesar Cipher
http://www.trincoll.edu/~cpsc/cryptography/caesar.html

Certicom
http://www.certicom.ca/

Certificate Authority Services
https://certs.netscape.com/client.html

Computer Data Authentication Federal Information Processing Standards Publication 113
http://www.nist.gov/itl/div897/pubs/fip113.htm

Computer Network Security
http://www.weru.ksu.edu/people/dudley/SECURITY/security.html#Eavesdropping

Computer Science Laboratory of SRI International
http://www.csl.sri.com/~gong/

Computer Science Laboratory of SRI International: Journal publications
http://www.csl.sri.com/~gong/papers/journal-pubs.html

Computer Security Roles of NIST and NSA
http://bilbo.isu.edu/security/csl/csl02-91.html

Connected: An Internet Encyclopedia
http://www.FreeSoft.org/CIE/index.htm

Cryptographic Algorithms
http://www.cs.hut.fi/ssh/crypto/algorithms.html

Cryptography
http://www.trincoll.edu/~cpsc/cryptography/index.html

Cryptography and Security Bookmarks
http://student.vub.ac.be/~vcolet/crypto.html

Cryptosystems: misc
http://www.ioc.ee/home/helger/crypto/htmls/systems.html

Cryptology
http://www.fas.org/irp/wwwsigin.html#crypto

Cylink
http://www.cylink.com/

Department of Defense (DOD) Public Key Infrastructure (PKI)
http://www.disa.mil/ciss/pki.html

Digital Signature Guidelines Tutorial
http://www.abanet.org/scitech/ec/isc/dsg-tutorial.html

Gateway to Information Security
http://www.securityserver.com/cgi-local/ssis.pl/category/encry8.htm#Cryptographic

Gene Tsudik: Security Publications
http://www.isi.edu/~gts/pubs.html

How Can the Network Be Made Secure?
http://www.sun.com/security/wp-vpn.tco/chap6.html

IBM Security Architecture
http://www.rs6000.ibm.com/resource/aix_resource/Pubs/redbooks/htmlbooks/sg244579.00/4579fm.html

IBM security home page: SecureWay
http://www.ibm.com/security/

IBM Software Glossary
http://www.networking.ibm.com/nsg/nsgmain.htm

IEEE Cipher
http://www.itd.nrl.navy.mil/ITD/5540/ieee/cipher/

Info-Sec.Com - resource on Information Security in its many guises.
http://www.info-sec.com/

International Computer Security Association (ISCA) Information Library
http://www.ncsa.com/library/library.html

Internet Protocol Next Generation (IP ng)
http://ganges.cs.tcd.ie/4ba2/ipng/

IP Next Generation (IPng)
http://playground.sun.com/pub/ipng/html/ipng-main.html

MING IPv6 Overview
http://www.mentat.com/Documentation/white_papers/Ming_overview.html

Motorola Information Security Division (ISD): Key Management
http://www.mot.com/GSS/SSTG/ISD/Keymanagement/index.html

National Institute of Standards and Technology (NIST)

Computer Security Resource Clearinghouse (CSRC)
http://www-08.nist.gov/

National Security Agency (NSA)
http://www.fas.org/irp/nsa/index.html

Navy INFOSEC WebSite
http://infosec.nosc.mil/TEXT/EKMS/

Network Secure Communications
http://xfactor.wpi.edu/Works/MQP/securenet/root/root.html

Network Security International Association's (NetSec Int'l) Web site
http://www.netsec-intl.com/

Network Security Threats
http://csrc.ncsl.nist.gov/nistpubs/800-7/node113.html

Next Generation TCP BOF (TCPNG)
Reported by Robert Braden/USC Information Sciences Institute
ftp://ftp.nordu.net/ietf/94dec/tcpng-minutes-94dec.txt

NIST Advanced Authentication Technology Program
http://csrc.ncsl.nist.gov/authentication/

NIST Computer Security Division (893)
http://www.itl.nist.gov/div893/

NIST: Other Security Publications
http://csrc.nist.gov/secpubs/

Overview on IPv6
http://www.seas.gwu.edu/student/reto/ipv6/overview.htm

Publications on Proofs of Human Knowledge
http://world.std.com/~dpj/links.html

RFC 1949 Scalable Multicast Key Distribution
http://www.kashpureff.org/nic/rfcs/1900/rfc1949.txt.html

Ronald L. Rivest: Algorithms, Protocols, Etc.
http://theory.lcs.mit.edu/~rivest/crypto-security.html#Algorithms

Ronald L. Rivest: Cryptography and Security
http://theory.lcs.mit.edu/~rivest/crypto-security.html

RMTP Security
http://www.trl.ibm.co.jp/rmtp/proms96.psz

Secure distributed computing
http://www.research.att.com/~reiter/#Metrics

Security in Open Systems
http://csrc.ncsl.nist.gov/nistpubs/800-7/main.html

Security Protocols and Services
http://www.rsa.com/rsalabs/newfaq/secprserv.htm

Special Pub 800-12 -- An Introduction to Computer Security: The NIST Handbook
http://sunsite.rediris.es/ftp/pub/security/docs/nistpubs/800-12/

System Authentication Design with PKE
http://xfactor.wpi.edu/Works/MQP/securenet/root/node58.html

Technical Overview of PEKE (Probabilistic Encryption Key Exchange)
http://www.connotech.com/PEKEDESC.HTM

The Computer Privacy Handbook : A Practical Guide to E-Mail Encryption, Data Protection, and Pgp Privacy Software
http://www.amazon.com/exec/obidos/ISBN%3D1566091713/gatewaytoinformaA/002-3861940-1470448

Transition from IPv4 to IPv6
http://www.tascomm.fi/~jlv/ngtrans/

Università di Torino Security Group
http://maga.di.unito.it/security/home.html

University of Texas: Networking Research Laboratory
http://www.cs.utexas.edu/users/lam/NRL/

University of Texas: Network Security
http://net.cs.utexas.edu/users/lam/NRL/network_security.html

U.S. Department of Defense Security Options for the Internet Protocol Request for Comments: 1108
http://andrew2.andrew.cmu.edu/rfc/rfc1108.html

VENONA Documents
http://www.nsa.gov:8080/docs/venona/venona.html

VeriSign
https://digitalid.verisign.com/

Virtual Private Network (VPN) Source Page
http://techweb.cmp.com/internetwk/VPN/

Whitepaper: The Kerberos Authentication Service
http://www.suite.com/whitepapers/wp5.html#2_1

Yvo G. Desmedt: cryptography, network security, and computer security.
http://www.cs.uwm.edu/faculty/desmedt/index.html

Zero-Knowledge and Secure Function Computation
http://theory.lcs.mit.edu/~oded/homepage.html

APPENDIX B. INTERNET WEB-BASED SEARCH ENGINES

The Internet search engines listed below were used to find the security-related Web sites listed in Appendix A.

AltaVista Technology, Inc.
http://www.altavista.com/

Excite Home
http://www.excite.com/

HotBot
http://www.hotbot.com/IU0JNEXRAD4E629912213839EE46F9D75075BA49/index.html

Infoseek
http://www.infoseek.com/Home?pg=Home.html&sv=N3

Yahoo!
http://www.yahoo.com/

LIST OF REFERENCES

Bay Networks, Inc., "IPv6", *Bay Networks Whitepaper*, 1997.

Certicom, Corp., "Current Public-Key Cryptographic Systems", *Certicom whitepaper*, 1997.

Federal Standard 1037C: Glossary of Telecommunication Terms, http://www.its.bldrdoc.gov/fs-1037/

Fisher, S. E., "VPNs Use Tunneling To Build Private Business Links", *Datamation*, June 1996.

Ford, W., *Computer Communications Security*, Prentice-Hall, Inc., 1994.

FTP Software Inc., "FTP Software and Intranet Security: What the IT Manager Needs to Know", *FTP Software Whitepaper*, 1997. http://www.ftp.com/product/whitepapers/secure.htm

Gong, L., and Shacham, N., "Multicast Security and Its Extension to a Mobile Environment", *ACM-Baltzer Journal of Wireless Networks*, 1(3):281-295, 1995.

Kaufman, C., Perlman, R., and Speciner, M., Network Security: Private Communication in a Public World, Prentice-Hall, Inc., 1995.

Jacobson, V., McCanne, S., and Floyd, S., "A Privacy Architecture for Lightweight Sessions", *ARPA Network PI Meeting*, Santa Fe, NM, 1994.

Jane's Internet Defense Glossary
http://www.janes.com/defence/resources/defres_gloss.html

Lin, J. C., and Paul, S., "RMTP: A Reliable Multicast Transport Protocol", *Proceeding of IEEE INFOCOM '96, pp. 1414-1424*, 1996.

Lundy, G. M., Sanjoy, P., and Dismuke, J., "A Formal Model of Reliable Multicast Transport Protocol", 1996.

Menezes, A. J., van Oorschot, P.C., and Vanstone, S.A., *Handbook of Applied Cryptography, CRC Press, 1996.*

Petitt, D. G., "Solutions for reliable Multicasting, Master's Thesis", *Naval Postgraduate School*, September, 1996.

Rao, R., and Neer, M., "HIPNET User Requirements", 28 November 1997, Space and Naval Systems Center, Code D82, San Diego, CA.

Russell, D., and Gangemi, G.T., *Computer Security Basics*, O'Reilly & Associates, Inc., 1991.

Schneier, B., *Applied Cryptography Second Edition: Protocols, Algorithms and Source Code in C*, John Wiley & Sons, Inc., 1996.

Shiroshita, T., Sano, T., Takahashi, O., Yamashita, M., Yamanouchi, N., and Kushida, T., "Performance Evaluation of Reliable Multicast Transport Protocol for Large-scale Delivery", *IFIP Fifth International Workshop on Protocols For High-Speed Networks PfHSN '96)*, Sophia Antipolis, France, 1996.

Stallings, W., *Network and Internetwork Security: Principles and Practice*,.

Tanenbaum, A. S., *Computer Networks*, Prentice-Hall, Inc., 1996.

TechWeb Technology Encyclopedia,
http://www.techweb.com/encyclopedia/defineterm.cgi

GLOSSARY

Technical terms used in this thesis are included in the glossary below. More detailed definitions, or related terms not included here, may be found in the following suggested sources:

Federal Standard 1037C: Glossary of Telecommunication Terms,

http://www.its.bldrdoc.gov/fs-1037/

MITRE Security Glossary

http://www.mitre.org:80/resources/centers/infosec/publications/sec-glossary/

TechWeb Technology Encyclopedia,

http://www.techweb.com/encyclopedia/

Access Control List (ACL) - A set of data containing names, passwords, or other forms of user/group identification with which to control and manage network resources (i.e., associated files, directories, etc.).

Asymmetric Key - Also called public key, a key pair (a private and an associated public key) is used to assure the confidentiality of the plain (message) text. Plain text is encrypted into cipher text, via the receiver's published public key, and cipher text is decrypted into plain text, via use the receiver's unpublished private key.

Authentication - A property of a communication, or information security system, which establishes the identity of participating parties to each other. Authentication can be one-way, (sender authenticated to receiver), or mutual (sender authenticated to receiver and vice versa).

Block Cipher - A reversible function, such that the operation of the cipher on a message block (typically 64 bits) with a secret key produces a block of cipher text. Usually the message and cipher text share the same alphabet, so the block cipher permutes the message in a key-specified way. There should be no (efficient) way to deduce the key, given any number of message/cipher text pairs, and no way to deduce the block cipher function from the message block, or the message block from the block cipher function, without the key.

Brute Force Attack - A method to defeat the security protection provided by an encryption key. This attack tries "all possible keys" to convert cipher text into clear text. For a 56 bit key there are 7.2 x 10^16 possible keys, for a 128 bit key, there 3.4 x 10^38 possible keys, a daunting number.

Conference Keying - A generalized form of two-party key establishment schemes that provides all conference nodes with a shared secret key.

Confidentiality - A property of a communication, or information security system, which ensures participating parties that the contents of information they convey to each other will not become known, or available to individuals, processes, or entities lacking authorized access.

Digital Signature - The concept of digital signatures (using public keys) was first proposed by Diffie and Hellman in 1976. A digital signature may utilize public-key encryption, or secret-key encryption (with/without hashing techniques). This digest primitive facilitates authentication services, by computing and attaching a condensed text string, uniquely associated with a particular sender/entity, to a message.

Digital Signature Standard (DSS) - A Digital Signature Algorithm (DSA) proposed in 1991 by the U.S. National Institute of Standards (NIST) as a U.S. Federal Information Processing Standard (FIPS 186 Digital Signature Standard), a variation of the ElGamal scheme.

Encryption - A processes that enciphers (i.e., transforms) messages (or information) into random sequences devoid of intelligent information.

Firewall - A device(s) to filter/sanitize incoming and outgoing network message traffic (i.e, secure a specified network). Firewalls can be individually server (router), or

client (host) based, or used in combination to provide network protection. Examples are: Packet Filter (IP address and/or port number message blocking, "screening router"), Proxy Server (a relay between two networks), Network Address Translation (NAT) (presents one IP address to the outside world hiding IP addresses of internal network client/host nodes), and Stateful Inspection (tracks transactions to verify that inbound packet destination matches previous outbound request source).

Hash Function - An algorithm that produces a fixed-size (usually shorter), unique output (hash) value from a variable-sized input message. Hash functions are used in generating digital signatures.

Integrity - A property of a communication, or information security system, which ensures participating parties that the data contents of information they convey to each other has not unknowingly been corrupted, changed, or tampered in transit, storage, or use.

Key - A special esoteric value used to easily unlock intelligent information from apparent gibberish. Two key types are in common use: secret (symmetric) and public (assymetric). Generally, longer key lengths, result in stronger security protection.

Message Digest (MD) - A condensed message version that is generated from a text message using a one-way hash function. Message digests can be used to create digital signatures.

Message Authentication Code (MAC) - A checksum generated when a secret key value is appended with a message block and passed through a one-way hash function, also known as a data authentication code (DAC). MACs/DACs are techniques used to authenticate messages (they also can verify integrity of file transfers). Similar to digital signatures, except that a secret key was used in their creation rather than a private key.

Multicast - The transmission of data originating from a single source to a specified group of network receivers.

Multiple Level Security (MLS) - A class of automated information system (AIS) containing data with different sensitivities that simultaneously permits access by users with different security clearances and needs-to-know, but prevents users from obtaining access to information for which they lack authorization. [Nat'l Computer Security Center, Trusted Network, Glossary of Computer Security Terms, NCSC-TG-004, Oct. 1988.]

Nonce - A non-repeated number (e.g., large random number, sequence number, or timestamp) used in a security protocol (ensures challenge unpredictability) to protect against active masquerade attacks.

Non-repudiation - Non-repudiation of origin ensures that the sender can not deny authoring a particular document, much like the seal of a Notary public. Similarly, non-repudiation of delivery verifies that the receiver can not deny receipt of a document(s).

Reflection Attack - An active security attack, initiated by an individual masquerading as a legitimate member of a data exchange. The masquerader replays an exchange from one protocol in a second protocol to become authenticated as a legitimate user.

Secret Sharing Scheme - A multi-party protocol, related to key establishment, to protect against key loss, or compromise. Implementations allow shared control and distributed/cooperative secret agreement for multiple sets of user groups. Simple two-party shared secret schemes use modular addition to generate the secret value.

Stream Cipher - A function that operates on single characters (i.e., small block sizes), or bits, and is capable of higher throughputs than other encryption techniques. A stream cipher generates a key-stream and

encryption results from combining this key-stream with the plain text, usually via a bit-wise exclusive-OR operation. Key-stream generation can be independent of the plain and cipher text (yielding a synchronous stream cipher, the most common form), or it can depend on the data and the cipher text (i.e., a self-synchronizing stream cipher). Stream ciphers are well suited to hardware implementation. Software implementations tend to suffer performance reductions resulting from increased computational demands.

Substitution - A replacement of each character (or character group) of message (plain) text with a predetermined character (or character group) to form the cipher text. The cipher text can be from the same alphabet (mono-alphabetic) as the plain text, or can be changed for each text character (polyalphabetic substitution) of the message on a predetermined schedule.

Symmetric Key - Also called secret key, the same key is used to encrypt a plain (message) text into cipher text, and decrypt cipher text into plain text. The strength of the encryption scheme is directly dependent upon the secrecy of this key.

Threshold Scheme - A form of secret sharing that only requires a specified number (vice all) of network user nodes

to reconstruct the network secret (key). Threshold schemes are a specialized type of generalized secret sharing.

Timestamp - A type of nonce that is used to deter the effectiveness of reflection attacks on security protocols.

Transposition - A reordering of the characters in a plain text block (an unencrypted text string) through the operation of a permutation function.

Trusted Third Party (TTP) - A centralized server based scheme to reduce the number of keys required (n x [n-1]) to provide secure mediated authentication services to all network user nodes (n). Authentication servers (AS), key distribution centers (KDC), certification authorities (CA), and key translation centers (KTC) are examples of trusted third parties.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center 2
   8725 John J. Kingman Rd., STE 0944
   Ft. Belvoir, VA 22060-6218

2. Dudley Knox Library 2
   Naval Postgraduate School
   411 Dyer Rd.
   Monterey, CA 93943-5101

3. Dr. Dan Boger, Chairman 1
   Department of Computer Science, Code CS
   Naval Postgraduate School
   833 Dyer Rd.
   Monterey, CA 93943

4. Professor Bert Lundy 1
   Department of Computer Science, Code CS/LN
   Naval Postgraduate School
   833 Dyer Rd.
   Monterey, CA 93943

5. COMMANDING OFFICER 1
   ATTN BOB KOCHANSKI
   SPAWAR SYSTEMS CENTER San Diego D80
   53560 HULL ST
   SAN DIEGO CA 92152-5001

6. COMMANDING OFFICER 1
ATTN EUGENA ENGH
SPAWAR SYSTEMS CENTER San Diego D87
53560 HULL ST
SAN DIEGO CA 92152-5001

7. COMMANDING OFFICER 1
ATTN MIKE HARRISON
SPAWAR SYSTEMS CENTER San Diego D872
53560 HULL ST
SAN DIEGO CA 92152-5001

8. COMMANDING OFFICER 1
ATTN SHARON KEITH
SPAWAR SYSTEMS CENTER San Diego D873
53560 HULL ST
SAN DIEGO CA 92152-5001

9. COMMANDING OFFICER 1
ATTN NATE KUNES
SPAWAR SYSTEMS CENTER San Diego D873
53560 HULL ST
SAN DIEGO CA 92152-5001

10. COMMANDING OFFICER 2
ATTN PHIL BARLOW
SPAWAR SYSTEMS CENTER San Diego D873
53560 HULL ST
SAN DIEGO CA 92152-5001